目录

第 1 章

控制系统基本理论

在这一章里，我们简要地介绍了控制系统的模型、时域分析方法、频域分析方法、极点配置与观测器设计及最优控制系统设计等内容。相应地，在第 3 章以大量的示例，说明利用 MATLAB 进行系统分析与设计的方法。

1.1 控制系统模型

系统的表示可用三种模型：传递函数、零极点增益、状态空间。每种模型均有连续/离散之分，它们各有特点，有时需在各种模型之间进行转换。

1.1.1 连续系统

1. 传递函数模型

$$H(s)=\frac{num(s)}{den(s)}=\frac{b_1s^m+b_2s^{m-1}+\cdots+b_{m+1}}{a_1s^n+a_2s^{n-1}+\cdots+a_{n+1}}$$

在 MATLAB 中，直接用分子/分母的系数表示，即

```
num=[b1,b2,…,bm];
den=[a1,a2,…,an];
```

2. 零极点增益模型

$$H(s)=k\frac{(s-z_1)(s-z_2)\cdots(s-z_m)}{(s-p_1)(s-p_2)\cdots(s-p_n)}$$

在 MATLAB 中，用[z,p,k]矢量组表示，即

```
z=[z1,z2,…,zm];
p=[p1,p2,…,pn];
k=[k];
```

3. 状态空间模型

$$\begin{cases}\dot{x}=ax+bu\\ y=cx+du\end{cases}$$

在 MATLAB 中，系统可用(a,b,c,d)矩阵组表示。

1.1.2 离散系统

1. 传递函数模型

$$H(z)=\frac{b_1z^m+b_2z^{m-1}+\cdots+b_{m+1}}{a_1z^n+a_2z^{n-1}+\cdots+a_{n-1}}$$

2. 零极点增益模型

$$H(z)=k\frac{(z-z_1)(z-z_2)\cdots(z-z_m)}{(z-p_1)(z-p_2)\cdots(z-p_n)}$$

3. 状态空间模型

$$\begin{cases}x(k+1)=ax(k)+bu(k)\\ y(k+1)=cx(k+1)+du(k+1)\end{cases}$$

1.1.3 模型之间的转换

同一个系统可用三种不同的模型表示，为分析系统的特性，有必要在三种模型之间进行转换。MATLAB 的信号处理和控制系统工具箱中，都提供了模型变换的函数：ss2tf,ss2zp,tf2ss,tf2zp,zp2ss,zp2tf，它们的作用可用图 1.1 所示的结构来表示。

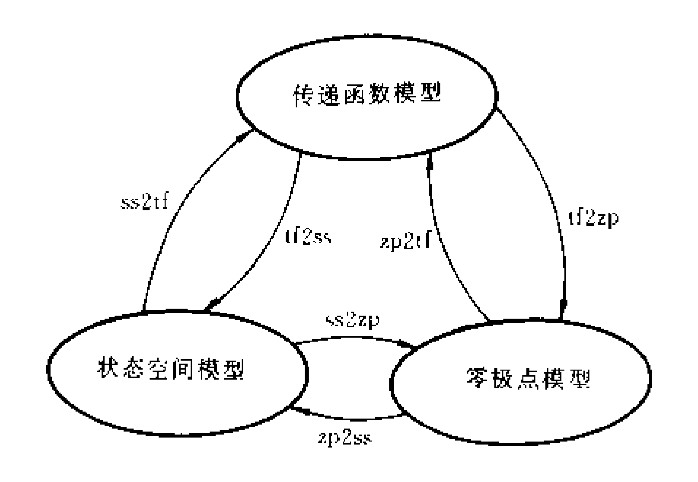


图 1.1 三种模型之间的转换

1.1.4 系统建模

对简单系统的建模可直接采用三种基本模型：传递函数、零极点增益、状态空间模型。但实际中经常遇到几个简单系统组合成一个复杂系统。常见形式有：并联、串联、闭环及反馈等连接。

1. 并联

将两个系统按并联方式连接，在 MATLAB 中可用 parallel 函数实现，详见第 2 章。

2. 串联

将两个系统按串联方式连接，在 MATLAB 中可用 series 函数实现，详见第 2 章。

3. 闭环

将系统通过正负反馈连接成闭环系统，在 MATLAB 中可用 cloop 函数实现，详见第 2 章。

4. 反馈

将两个系统按反馈方式连接成闭环系统，在 MATLAB 中可用 feedback 函数实现，详见第 2 章。

1.1.5 能控性和能观性

系统的能控性和能观性是两个重要的概念，是设计控制器和状态估计器的基础。

能控性是指系统状态在输入作用下转移到指定状态的能力，而能观性是指从输出中确定系统状态的能力。

对 n 阶系统

$$\begin{cases} \dot{x}=ax+bu \\ y=cx+du \end{cases}$$

能控性可借助于可控性矩阵

$$cam=[b,ab,a^2b,\cdots,a^{n-1}b]$$

当 rank(cam)=n，则系统能控，否则系统不能控。

同样，可构造能观性矩阵

$$oam=\begin{bmatrix} c \\ ca \\ ca^2 \\ \vdots \\ ca^{n-1} \end{bmatrix}$$

当 rank(oam)=n，则系统能观，否则系统不能观。

对离散系统有类似的结果。

在 MATLAB 中，可利用 ctrb 和 obsv 函数求出可控性和可观性矩阵，从而确定系统的可控性和可观性。

1.1.6 稳定性分析

对连续系统，如果系统的所有极点都位于左半 s-平面，即 $Re(p_i)<0$，$i=1,2,\cdots,n$，则系统是稳定的；否则系统是不稳定的。如果稳定系统的所有零点都位于左半 s-平面，即 $Re(z_i)<0$，$i=1,2,\cdots,m$，则称系统是最小相位的。

对离散系统，如果系统的所有极点都位于 z-平面的单位圆内，即 $|p_i|<1$，$i=1,2,\cdots,n$，则系统是稳定的；否则系统不稳定。如果稳定系统的所有零点都位于 z-平面的单位圆内，即 $|z_i|<1$，$i=1,2,\cdots,m$，则称系统是最小相位的。

在 MATLAB 中，可利用 pzmap 和 zplane 函数绘制出连续/离散系统的零极点图，从而确定系统的稳定性。但有些情况下，仍需要借助于求解 Lyapunov 方程，利用 Lyapunov 稳定性理论来判定系统的稳定性。在 MATLAB 中，利用 lpap 函数可求解 Lyapunov 方程。

1.1.7 系统模型的连续化与离散化

将连续系统模型经取样保持、A/D 变换后，可得到等效的离散系统，其中取样时间应

小于 Nyquist 采样时间，根据输入端采用的方法：零阶保持、一阶保持、双线性变换(Tustin 算法)、零极点匹配等方法，可得到不同的等效离散系统，这在 MATLAB 中可由 c2d 和 c2dm 函数得到。

离散系统的连续化是上述过程的逆过程，在 MATLAB 中可由 d2c 和 d2cm 函数得到。

1.2 控制系统的时域分析

系统仿真实质上就是对系统模型的求解，对控制系统来说，一般模型可转化成某个微分方程或差分方程表示，因此在仿真过程中，一般以某种数值算法从初态出发，逐步计算系统的响应，最后绘制出系统的响应曲线，这样可分析系统的性能。

控制系统最常用的时域分析方法是，当输入信号为单位阶跃和单位冲激函数时，求出系统的输出响应，分别称为单位阶跃响应和单位冲激响应。在 MATLAB 中，提供了求取连续系统的单位阶跃响应函数 step，单位冲激响应函数 impulse，零输入响应函数 initial 及任意输入下的仿真函数 lsim，相应的离散系统有函数：dstep，dimpulse，dinitial 和 dlsim，详见第 2 章。

1.3 控制系统的根轨迹

根轨迹法是分析和设计线性定常控制系统的图解方法，使用十分简便。特别是适用于多回路系统的研究，应用根轨迹比其它方法更为方便。

所谓根轨迹是指，当开环系统某一参数从零变到无穷大时，闭环系统特征方程的根在 s 平面上的轨迹。一般来说，这一参数选作开环系统的增益 k，而在无零极点对消时，闭环系统特征方程的根就是闭环传递函数的极点。

通常来说，要绘制出系统的根轨迹是很繁琐很难的事，因此在教科书中经常以简单系统的图示解法得到。但在现代计算机技术和软件平台的支持下，绘制系统的根轨迹变得轻松自如了。在 MATLAB 中，专门提供了绘制根轨迹有关的函数：rlocus，rlocfind，pzmap 等，详见第 2 章。

为说明根轨迹的作用，先绘制出简单二阶开环系统

$$H(s)=\frac{k}{s(0.5s+1)}$$

的根轨迹，这可在 MATLAB 中输入

```
num=[1];
den=[0.5  1  0];
rlocus(num,den)
```

则执行后可得如图 1.2 所示的根轨迹。

以图 1.2 为例，说明利用根轨迹来分析系统的各种性能。

1. 稳定性

当开环增益 k 从零变到无穷大时，图 1.2 中的根轨迹不会越过虚轴进入右半 s-平面，

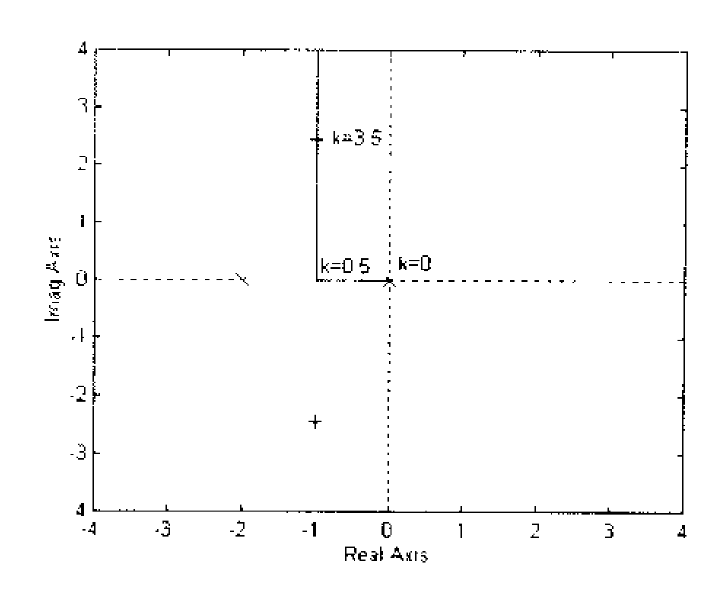


图 1.2 二阶系统的根轨迹

因此这个系统对所有的 k 值都是稳定的。如果根轨迹越过虚轴进入右半 s-平面，则其交点的 k 值就是临界开环增益，这可借助于函数 rlocfind 找出临界开环增益。

2. 稳态性能

开环系统在坐标原点有一极点，因此根轨迹上的 k 值就是静态速度误差系数。如果给定系统的稳态误差要求，则可由根轨迹确定闭环极点容许的范围。

3. 动态性能

当 $0<k<0.5$ 时，所有闭环极点位于实轴上，系统为过阻尼系统，单位阶跃响应为非周期过程；当 $k=0.5$ 时，闭环两个实数极点重合，系统为临界阻尼系统，单位阶跃响应仍为非周期过程，但速度更快；当 $k>0.5$ 时，闭环极点为复数极点，系统为欠阻尼系统，单位阶跃响应为阻尼振荡过程，且超调量与 k 成正比。

在求出系统根轨迹后，可对系统的性能有一定的了解，因此对系统分析是很有用的。对于设计系统可通过修改设计参数，使闭环系统具有期望的零极点分布，因此根轨迹对系统设计具有指导意义。

1.4 控制系统的频域分析

频域分析法是应用频率特性研究控制系统的一种经典方法。采用这种方法可直观地表达出系统的频率特性，分析方法比较简单，物理概念比较明确，对于诸如防止结构谐振、抑制噪声、改善系统稳定性和暂态性能等问题，都可以从系统的频率特性上明确地看出其物理实质和解决途径。

频率分析法主要包括三种方法：Bode 图(幅频/相频特性曲线)、Nyquist 曲线、Nichols 图。

1. Bode(波特)图

设已知系统的传递函数模型

$$H(s)=\frac{b_1s^m+b_2s^{m-1}+\cdots+b_{m+1}}{a_1s^n+a_2s^{n-1}+\cdots+a_{n+1}}$$

则系统的频率响应可直接求出

$$H(j\omega)=\frac{b_1(j\omega)^m+b_2(j\omega)^{m-1}+\cdots+b_{m+1}}{a_1(j\omega)^n+a_2(j\omega)^{n-1}+\cdots+a_{n+1}}$$

系统的 Bode 图就是 $H(j\omega)$的幅值与相位对 ω 进行绘图，因此也称为幅频和相频特性曲线。

在 MATLAB 中，可利用 bode 和 dbode 绘制连续和离散系统的 Bode 图，这大大简化了以往繁琐的作图过程。

2. Nyquist(奈奎斯特)曲线

Nyquist 曲线是根据开环频率特性在复平面上绘出幅相轨迹，根据开环的 Nyquist 曲线，可判断闭环系统的稳定性。

反馈控制系统稳定的充要条件是，Nyquist 曲线按逆时针包围临界点$(-1,j0)$ p 圈，p 为开环传递函数位于右半 s-平面的极点数，否则闭环系统不稳定。这就是著名的奈氏判据。当开环传递函数包含虚轴上的极点时，闭合曲线应以 $\varepsilon\rightarrow0$ 的半圆从右侧绕过该极点。

对离散系统，可得到类似的奈氏判据。

在 MATLAB 中，可利用函数 nyquist 和 dnyquist 绘出连续和离散系统的 Nyquist 曲线。

3. Nichols(尼柯尔斯)图

根据闭环频率特性的幅值和相位可作出 Nichols(图)，因此从中可直接得到闭环系统的频率特性。

在 MATLAB 中，可利用函数 nichols 和 dnichols 绘出连续和离散系统的 Nichols 图。

1.5 极点配置和观测器设计

给定控制系统，通过设计反馈增益阵 k 使闭环系统具有期望的极点，从而达到适当的阻尼系数和无阻尼自然频率，这就是极点配置问题。但极点配置是基于状态反馈，即 $u=-kx$，因此状态 x 必须可测，当状态不可测时，则应设计状态观测器。设计的状态观测器也应具有适当的频率特性，因此也可指定其极点位置，从而使状态观测器的设计转化为极点配置问题。

1.5.1 连续系统的极点配置

考虑控制系统

$$\dot{x}=ax+bu$$

选取

$$u=-kx$$

使闭环系统具有期望的极点 $\mu_1, \mu_2, \cdots, \mu_n$，关键是求出增益阵 k。

可以证明，对给定系统可进行任意极点配置的充要条件是系统状态完全可控。

极点配置有两种方法：第一种方法是采用变换矩阵 T，使系统具有期望的极点，从而求出矩阵 k；第二种方法基于 Caylay - Hamilton 理论，通过矩阵特征多项式 Φ(a)，可求出 k(这种方法称为 Ackermann 公式)。

1. 变换法

定义变换矩阵 T

$$T = cam * w$$

其中 cam 为可控性矩阵

$$cam = [b \quad ab \quad \cdots \quad a^{n-1}b]$$

w 阵定义为

$$w = \begin{bmatrix} a_{n-1} & a_{n-2} & \cdots & a_1 & 1 \\ a_{n-2} & a_{n-3} & \cdots & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ a_1 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix}$$

$a_i (i = 1, 2, \cdots, n-1)$ 为特征多项式系数，即

$$|sI - a| = s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n = 0$$

注意，这里 a 为系统矩阵，而 a_i 为标量。

定义新的状态 $\hat{x}$

$$x = T\hat{x}$$

由于系统可控，因此 T 可逆，系统变换成

$$\dot{\hat{x}} = T^{-1}aT\hat{x} + T^{-1}bu$$

其中

$$T^{-1}aT = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_1 \end{bmatrix} \qquad T^{-1}b = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$$

这就是能控标准型。令

$$\hat{k} = kT = [\delta_n \quad \delta_{n-1} \quad \cdots \quad \delta_1]$$

这时 $u = -\hat{k}\hat{x} = -kT\hat{x}$，因此系统方程可写成

$$\dot{\hat{x}} = T^{-1}aT\hat{x} - T^{-1}bkT\hat{x}$$

特征方程为

$$|sI - T^{-1}aT + T^{-1}bkT| = s^n + (a_1 + \delta_1)s^{n-1} + \cdots + (a_{n-1} + \delta_{n-1})s + (a_n + \delta_n) = 0$$

而由期望极点得

$$(s - \mu_1)(s - \mu_2)\cdots(s - \mu_n) = s^n + \alpha_1 s^{n-1} + \cdots + \alpha_{n-1}s + \alpha_n$$

比较两式得

$$\begin{cases} a_1+\delta_1=\alpha_1 \\ a_2+\delta_2=\alpha_2 \\ \vdots \\ a_n+\delta_n=\alpha_n \end{cases}$$

因此

$$k=\hat{k}T^{-1}=[\alpha_n-a_n \quad \alpha_{n-1}-a_{n-1} \quad \cdots \quad \alpha_1-a_1]\ T^{-1}$$

因此，极点配置的步骤为：

第 1 步　检查系统的可控性，当 rank(cam)＝n 时系统可控；在 MATLAB 中可用 rank (ctrb(a,b))实现；

第 2 步　确定系统矩阵 a 的特征多项式系数 a_i

$$|sI-a|=s^n+a_1s^{n-1}+\cdots+a_{n-1}s+a_n$$

在 MATLAB 中，可用 poly 函数实现；

第 3 步　确定变换矩阵 T

$$T=cam*w$$

第 4 步　确定期望特征多项式系数 α_i

$$(s-\mu_1)(s-\mu_2)\cdots(s-\mu_n)=s^n+\alpha_1s^{n-1}+\cdots+\alpha_{n-1}s+\alpha_n$$

在 MATLAB 中，可由 poly 函数实现；

第 5 步　求增益矩阵 k

$$k=[\alpha_n-a_n \quad \alpha_{n-1}-a_{n-1} \quad \cdots \quad \alpha_1-a_1]\ T^{-1}$$

2. Ackermann 公式

由上面讨论知，闭环系统

$$\begin{cases} \dot{x}=\hat{a}x \\ \hat{a}=a-bk \end{cases}$$

特征方程为

$$|sI-\hat{a}|=(s-\mu_1)(s-\mu_2)\cdots(s-\mu_n)=s^n+\alpha_1s^{n-1}+\cdots+\alpha_{n-1}s+\alpha_n$$

由此定义

$$\Phi(s)=s^n+\alpha_1s^{n-1}+\cdots+\alpha_{n-1}s+\alpha_nI$$

则对系统矩阵 a 有

$$\Phi(a)=a^n+\alpha_1a^{n-1}+\cdots+\alpha_{n-1}a+\alpha_n$$

根据 Cayley - Hamilton 理论，$\Phi(\hat{a})=0$ 成立，即

$$\Phi(\hat{a})=\hat{a}^n+\alpha_1\hat{a}^{n-1}+\cdots+\alpha_{n-1}\hat{a}+\alpha_nI=0$$

将 $\hat{a}=a-bk$ 代入，可得

$$k=[0 \quad 0 \quad \cdots \quad 0 \quad 1]\ cam^{-1}\Phi(a)$$

因此，极点配置步骤为：

第 1 步　检查系统的可控性，当 rank(cam)＝n 时，系统可控；

第 2 步　确定 Φ(・)系数 α_i

$$(s-\mu_1)(s-\mu_2)\cdots(s-\mu_n)=s^n+\alpha_1s^{n-1}+\cdots+\alpha_{n-1}s+\alpha_n$$

在 MATLAB 中，这可由 poly 函数实现；

第 3 步　求 Φ(a)，由于 a 为矩阵，因此在 MATLAB 中，这应由 polyvalm 函数实现；

第 4 步　求增益阵 k

$$k=[0\ \ 0\ \ \cdots\ \ 0\ \ 1]\,\mathrm{cam}^{-1}\Phi(a)$$

在 MATLAB 中，可很方便地利用上述两种方法进行极点配置设计，但利用 MATLAB 控制系统工具箱提供的 place 和 acker 函数进行极点配置设计，可免去繁琐的过程，因此一般可直接采用这两个函数进行设计。

1.5.2　连续系统的全阶状态观测器

考虑系统

$$\begin{cases}\dot{x}=ax+bu\\ y=cx\end{cases}$$

设系统完全可观，则可构造如图 1.3 所示的状态观测器。

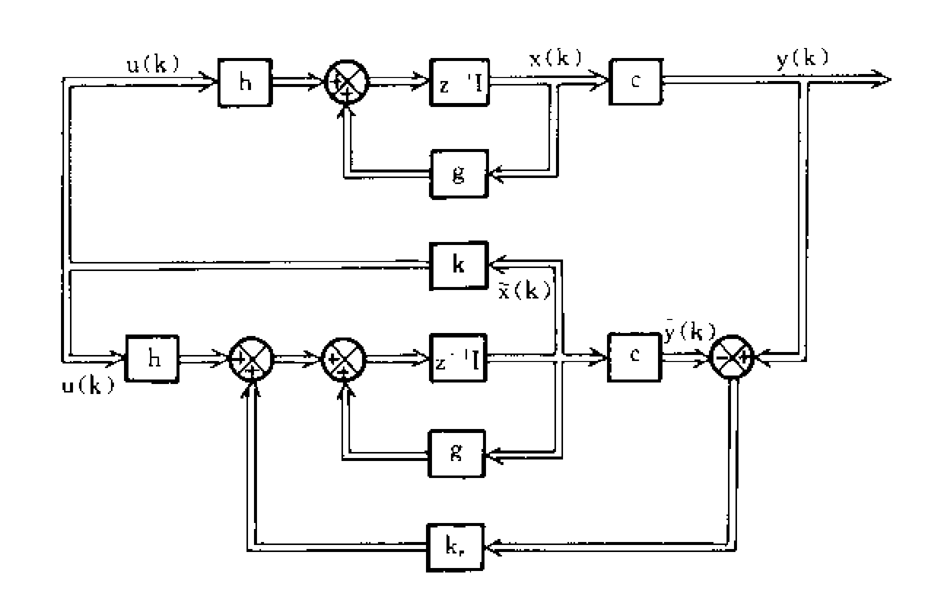


图 1.3　全阶状态观测器

为求出状态观测器的反馈增益阵 k_e，也可与极点配置类似的方法，有两种方法：第一种方法构造变换矩阵 Q，使系统变成标准能观型，然后根据特征方程求出 k_e；第二种方法可采用 Ackermann 公式

$$k_e=\Phi(a)\,\mathrm{oam}^{-1}[0\ \ 0\ \ \cdots\ \ 0\ \ 1]^{T}$$

其中 oam 为可观性矩阵。

利用对偶原理，可使设计问题大为简化。首先构造对偶系统

$$\begin{cases}\dot{p}=a'p+c'v\\ q=b'p\end{cases}$$

然后可由变换法或 Ackermann 公式求出极点配置的反馈增益矩阵 k，这也可以由 MATLAB 的 place 和 acker 函数得到；最后求出状态观测器的反馈增益

$$k_e=k^{T}$$

1.5.3　连续系统最小阶状态观测器

在系统中可精确地得到某些状态的测量值，因此无需进行全阶状态观测，故称之为降阶状态观测。当状态观测的阶降至最小时，则称最小阶状态观测。

假设状态矢量 x 为 n 维，输出 y 为 m 维，由于 y 可测量，因此只需对 n−m 个状态进行观测，这种结构如图 1.4 所示。

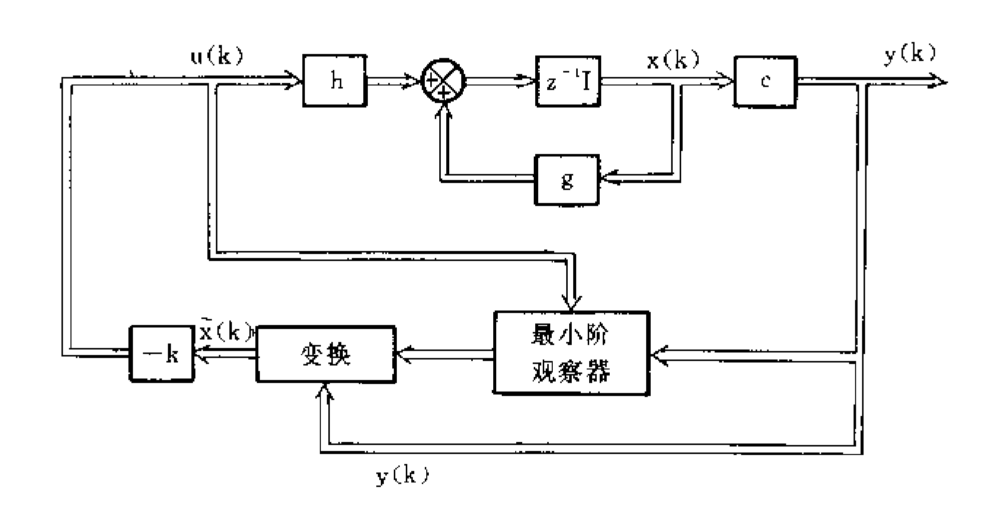


图 1.4　最小阶状态观测器

将状态 x 按直接可测和不可测分为 x_a 和 x_b，相应的系统方程写成分块矩阵的形式

$$\begin{cases}\begin{bmatrix}\dot{x}_a\\ \dot{x}_b\end{bmatrix}=\begin{bmatrix}a_{aa} & a_{ab}\\ a_{ba} & a_{bb}\end{bmatrix}\begin{bmatrix}x_a\\ x_b\end{bmatrix}+\begin{bmatrix}b_a\\ b_b\end{bmatrix}u\\ y=\begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}x_a\\ x_b\end{bmatrix}\end{cases}$$

可测部分

$$\dot{x}_a-a_{aa}x_a-b_au=a_{ab}x_b$$

其左边为可测，因此该式可作为输出方程。

不可测部分

$$\dot{x}_b=a_{bb}x_b+a_{ba}x_a+b_bu$$

这是新的状态方程。

与全阶状态器进行对比，可得到两者之间的对应关系，如表 1.1 所示。

表 1.1　全阶与最小阶状态观测器对比关系

全阶	最小阶	全阶	最小阶
x	x_b	y	$\dot{x}_a-a_{aa}x_a-b_au$
a	a_{bb}	c	a_{ab}
bu	$a_{ba}x_a+b_bu$	$k_e(n\times1)$	$k_e[(n-m)\times1]$

由此可得最小阶状态观测器

$$\begin{cases}\dot{\tilde{\eta}}=(a_{bb}-k_e a_{ab})\tilde{\eta}+[(a_{bb}-k_e a_{ab})k_e+a_{ba}-k_e a_{aa}]\,y+(b_b-k_e b_a)u \\ \eta=x_b-k_e y=x_b-k_e x_a \\ \tilde{\eta}=\tilde{x}_b-k_e y=\tilde{x}_b-k_e x_a\end{cases}$$

k_e 可根据表 1.1 按全阶状态观测器设计，k_e 为 $n-m$ 维矢量。

构造对比系统

$$\begin{cases}\dot{p}=a_c p+b_c v \\ q=c_c p\end{cases}$$

其中

$$a_c=a_{bb} \qquad b_c v=a_{ba}x_a+b_b u \qquad c_c=a_{ab}$$

然后，根据 MATLAB 的 place 和 acker 函数进行设计，得到反馈增益阵 k_e。

1.5.4 离散系统的极点配置

离散系统的极点配置和连续系统的极点配置基本相同，求解反馈增益阵 k 的方程实质上也是相同的。

离散系统

$$x(k+1)=g\,x(k)+h\,u(k)$$

在系统可控，且控制信号 u(k) 无界的情况下，选用反馈控制

$$u(k)=-k\,x(k)$$

可使闭环系统具有期望的极点 $\mu_i(i=1,2,\cdots,n)$。

闭环系统可写成

$$x(k+1)=(g-hk)x(k)$$

求解过程与连续系统一样，这里不再赘述。在 MATLAB 中，可直接采用控制工具箱中的 place 和 acker 进行设计。

但应说明一点，当闭环极点全选为 0 时，系统将产生无阻尼响应，这时只要控制 u(k) 无界，可使任何非零误差矢量至多在 n 步中降至零。当然调整时间取决于采样周期 T，当 $T\to 0$ 时，说明调整时间可为无限小，但控制信号 u(k) 也为无限大。我们一般希望调整时间越短越好，但这时会导致 u(k) 的饱和，一旦 u(k) 饱和，系统成为非线性系统，从而采用的设计方法失效。因此，应该在调整时间和 u(k) 幅值之间折衷考虑。

1.5.5 离散系统的状态观测器

离散系统状态观测器的设计与连续系统类似，也是借助于对偶原理来完成。

对离散系统

$$\begin{cases}x(k+1)=gx(k)+hu(k) \\ y(k)=cx(k)\end{cases}$$

构造对偶系统

$$\begin{cases}p(k+1)=g'p(k)+c'v(k) \\ q(k)=h'p(k)\end{cases}$$

然后利用 MATLAB 的 place 和 acker 函数求得增益阵 k，最后得状态观测器增益阵 $k_e=k'$。

全阶和最小阶状态观测器的设计也与连续系统类似，这里不再赘述。

1.6 最优控制系统设计

1.6.1 状态反馈的线性二次型最优控制器设计

线性时不变系统

$$\dot{x}(t)=ax(t)+bu(t)$$
$$y(t)=cx(t)+du(t)$$

线性二次型(LQ)最优控制器的任务是设计 u(t)，使线性二次型最优控制指标

$$J=\frac{1}{2}x^T(t_f)Sx(t_f)+\int_{t_0}^{t_f}[x^T(t)Qx(t)+u^T(t)Ru(t)]\,dt$$

最小。对最优控制而言，R 为对称正定阵，而 Q 为对称半正定阵。

在性能指标 J 中，第一项 $x^T(t_f)Sx(t_f)$表示稳态误差，第二项 $\int_{t_0}^{t_f}x^T(t)Qx(t)\,dt$ 表示总的暂态误差，第三项 $\int_{t_0}^{t_f}u^T(t)Ru(t)\,dt$ 表示暂态过程中所消耗的控制能量。

当 $x(t_f)$值固定时，则为终端控制问题，特别是当 $x(t_f)=0$ 时，则为调节器问题；当 t_0、t_f 均固定时，则为暂态过程最优控制。

最优控制设计问题，可根据变分法求得。首先构造 Hamilton 函数

$$H=-\frac{1}{2}[x^T(t)Qx(t)+u^T(t)Ru(t)]+\lambda^T(t)[ax(t)+bu(t)]$$

当 u(t)无界时，Hamilton 函数对 u(t)求导并令其值为 0，

$$\frac{\partial H}{\partial u}=-Ru(t)+b^T\lambda(t)=0$$

从而得最优控制

$$u(t)=R^{-1}b^T\lambda(t)$$

可以证明，λ(t)可由 $\lambda(t)=-p(t)x(t)$得到，而 p(t)满足微分 Riccati 方程

$$a^Tp(t)+p(t)a+Q-p(t)bR^{-1}b^Tp(t)=-\frac{d}{dt}p(t)$$

当 $t_f\to\infty$时，p(t)趋向于常值矩阵，即$(d/dt)p(t)=0$，因此 p(t)满足代数 Riccati 方程

$$a^Tp+pa+Q-pbR^{-1}b^Tp=0$$

由此得到的控制律为

$$u(t)=-k\,x(t)$$
$$k=R^{-1}b^Tp$$

LQ 控制器的结构框图如图 1.5 所示。

在 MATLAB 中，求解代数 Riccati 方程可采用 are 函数，线性二次型调节器的设计可直接采用 lqr 函数。

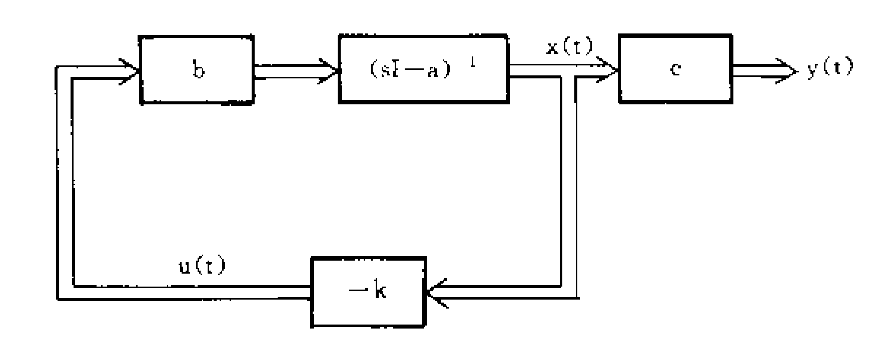


图 1.5 LQ 控制器结构

1.6.2 输出反馈的线性二次型最优控制

当采用输出反馈设计线性二次型最优控制时，只需对 LQ 算法稍作修正，设引入最优控制

$$u(t)=-k_0 y(t)=-k_0\,c\,x(t)$$

则最优反馈矩阵 k_0 应该使 $a-bk_0c$ 为渐近稳定矩阵，k_0 取

$$k_0=R^{-1}b^Tpzc^T(czc^T)^{-1}$$

其中 p 和 z 分别满足

$$p(a-bk_0c)+(a-bk_0c)^Tp+c^Tk_0{}^TRk_0c+Q=0$$

$$z(a-bk_0c)^T+(a-bk_0c)z+I_n=0$$

注意，在 MATLAB 中，输出反馈线性二次型最优控制器不能由 lqry 函数进行设计。

1.6.3 线性二次型 Guass(高斯)最优控制

设系统中含有噪声，测量时也含有噪声，即模型为

$$\begin{cases}\dot{x}(t)=ax(t)+bu(t)+\gamma w(t)\\ y(t)=cx(t)+v(t)\end{cases}$$

其中 w(t)的 v(t)均为白噪声，且有

$$E\{w(t)w^T(t)\}=W$$

$$E\{v(t)v^T(t)\}=V$$

$$E\{w(t)v^T(t)\}=0$$

则线性二次型 Guass(LQG)控制问题，是设计控制律使代价函数

$$J=\frac{1}{2}\lim_{t_f\to\infty}E\left\{\int_0^{t_f}[x^T(t)Qx(t)+u^T(t)Ru(t)]\,dt\right\}$$

最小。利用分离原理，这一问题可分成两个子问题：

· 利用卡尔曼滤波理论，从状态 x(t)中得到最优估计 $\hat{x}(t)$，使

$$E\{[x(t)-\hat{x}(t)]^T[x(t)-\hat{x}(t)]\}$$

最小。这种最优估计器为

$$\dot{\hat{x}}(t)=a\hat{x}(t)+bu(t)+k_l[y(t)-c\hat{x}(t)]$$

其中滤波器增益 k_l 为

$$k_l = pc^T V^{-1}$$

估计误差协方差阵 p 满足代数 Riccati 方程

$$ap + pa^T + \gamma W \gamma^T - pc^T V^{-1} cp = 0$$

· 利用卡尔曼状态估计值 $\hat{x}(t)$ 代替 x(t)，设计标准的确定性 LQ 控制律，即

$$u(t) = -k\hat{x}(t)V^{-1}$$

LQG 控制器的结构框图如图 1.6 所示。

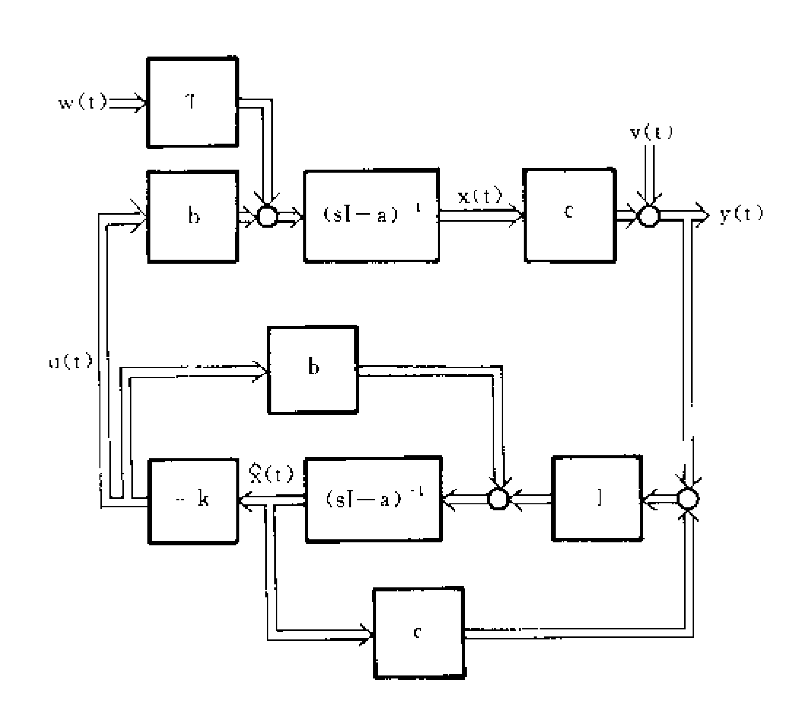


图 1.6　LQG 控制器结构

线性二次型控制器具有良好的稳定性和敏感性，但在实际实现时却遇到了困难，因为状态必须可测。当状态不可测时，可采用回路传输恢复技术来进行设计。

在图 1.5 所示的 LQ 系统中，开环传递函数为

$$k(sI-a)^{-1}b$$

而在图 1.6 所示的 LQG 系统中，相应的开环传递函数为

$$k(sI-a+bk+k_lc)^{-1}k_lc(sI-a)^{-1}b$$

设原系统是最小相位的，且选择代价函数 J 的加权阵 $R = rI$，$Q = q^2bb^T$，可以证明

$$\lim_{q\to\infty}\{k(sI-a+bk+k_lc)^{-1}k_lc(sI-a)^{-1}b\} = k(sI-a)^{-1}b$$

这可导出称作回路传输恢复技术的控制系统设计方法，它可分两步：

第 1 步　回路成形。取加权阵 $Q = c^Tc$，通过修改参数 r 设计 LQ 控制器，使得到的开环传递函数类似于目标传递函数，其敏感特性和补偿敏感特性都具有理想的形状；

第 2 步　恢复。增大参数 q，求解代数 Riccati 方程

$$ap + pa^T + \gamma W \gamma^T - pc^T V^{-1} cp = 0$$

直到 LQG 控制系统接近于 LQ 的性能。

1.6.4 离散系统线性二次型控制

离散系统为

$$x(k+1)=gx(k)+hu(k)$$

设计最优控制律 $u(k)=-k\,x(k)$，使性能指标

$$J=\frac{1}{2}\sum_{k=0}^{\infty}[x^T(k)qx(k)+u^T(k)ru(k)]$$

最小。

与连续系统类似的方法可得到

$$k=(r+h^Tph)^{-1}h^Tpg$$

$$p=q+g^Tpg-g^Tph(r+h^Tph)^{-1}h^Tpg$$

当然在实际设计时，可利用 MATLAB 提供的 dlqr 函数直接进行设计。

第2章

控制系统工具箱函数

MATLAB 包含了进行控制系统分析与设计所必须的工具箱函数，有关这些函数的使用可通过 Help 命令得到。为使用方便，本章将给出这些函数的使用说明。为方便用户查询，本节先简要地分组列出各种工具箱函数(如表 2.1～2.10 所示)，然后给出按字母顺序排列的索引(如表 2.11 所示)。

表 2.1　模 型 建 立

函数名	功　能
augstate	将状态增广到状态空间系统的输出中
append	两个状态空间系统的组合
parallel	系统的并联连接
series	系统的串联连接
feedback	两个系统的反馈连接
cloop	状态空间系统的闭环形式
ord2	产生二阶系统
rmodel，drmodel	稳定的随机 n 阶模型
ssdelete	从状态空间系统中删除输入、输出或状态
ssselect	从大状态空间系统中选择一个子系统
connect，blkbuild	将方框图转换为状态空间模型
estim，destim	生成连续/离散状态估计器或观察器
reg，dreg	生成控制器/估计器
pade	时延的 pade 近似

表 2.2 模型变换

函数名	功　　能
c2d, c2dt	将连续时间系统转换成离散时间系统
c2dm	将连续状态空间模型变换成离散状态空间模型
d2c	将离散时间系统变换成连续时间系统
d2cm	按指定方式将离散时间系统变换成连续时间系统
ss2tf	变系统状态空间形式为传递函数形式
ss2zp	变系统状态空间形式为零极点增益形式
tf2ss	变系统传递函数形式为状态空间形式
tf2zp	变系统传递函数形式为零极点增益形式
zp2ss	变系统零极点形式为状态空间形式
zp2tf	变系统零极点形式为传递函数形式

表 2.3 模型简化

函数名	功　　能
balreal, dbalreal	平衡状态空间的实现
minreal	最小实现性与零极点对消
modred, dmodred	模型降阶

表 2.4 模型实现

函数名	功　　能
canon	状态空间的正则形式转换
ctrbf, obsvf	可控性和可观性阶梯形式
ss2ss	相似变换

表 2.5 模型特性

函数名	功　　能
ctrb, obsv	可控性和可观性矩阵
gram, dgram	求可控和可观性 gram 矩阵
dcgain, ddcgain	计算系统的稳态(D. C.)增益
damp, ddamp	求衰减因子和自然频率
covar, dcovar	白噪声的协方差矩阵
esort, dsort	特征值排序
tzero	传递零点
printsys	显示或打印线性系统

表 2.6 方程求解

函数名	功　　能
are	代数 Riccati 方程求解
lyap, lyap2, dlyap	Lyapunov 方程求解

表 2.7　时域响应

函数名	功　　能
step	求连续系统的单位阶跃响应
dstep	求离散系统的单位阶跃响应
impulse	求连续系统的单位冲激响应
dimpulse	求离散系统的单位冲激响应
initial	求连续系统的零输入响应
dinitial	求离散系统的零输入响应
lsim	仿真任意输入的连续系统
dlsim	仿真任意输入的离散系统
ltitr	求线性时不变系统的时间响应

表 2.8　频域响应

函数名	功　　能
bode	求连续系统的 Bode 频率响应
dbode	求离散系统的 Bode 频率响应
nyquist	求连续系统的 Nyquist 频率曲线
dnyquist	求离散系统的 Nyquist 频率曲线
nichols	求连续系统的 Nichols 频率响应曲线
dnichols	求离散系统的 Nichols 频率响应曲线
ngrid	绘制 Nichols 曲线网络
sigma	求连续状态空间系统的奇异值 Bode 图
dsigma	求离散状态空间系统的奇异值 Bode 图
freqs	模拟滤波器的频率响应
freqz	数字滤波器的频率响应
margin	求增益和相位裕度
ltifr	求线性时不变响应

表 2.9　根轨迹

函数名	功　　能
pzmap	绘制系统的零极点图
rlocus	求系统根轨迹
rlocfind	计算给定根的根轨迹增益
sgrid	在连续系统根轨迹和零极点图中绘制阻尼系数和自然频率栅格
zgrid	在离散系统根轨迹和零极点图中绘制阻尼系数和自然频率栅格

表 2.10 估计器/调节器设计

函数名	功能
lqe, lqe2, lqew	连续系统线性二次型估计器设计
dlqe, dlqew	离散系统线性二次型估计器设计
lqed	根据连续代价函数进行离散估计器设计
lqr, lqr2, lqry	连续系统的线性二次型调节器设计
dlqr, dlqry	离散系统的线性二次型调节器设计
lqrd	根据连续代价函数进行离散调节器设计
place, acker	极点配置增益选择

表 2.11 函数索引表

函数名	页码	函数名	页码	函数名	页码
append	20	dstep	55	pade	34
are	52	dnyquist	67	parallel	21
augstate	20	esort, dsort	49	place, acker	86
balreal, dbalreal	41	estim, destim	31	printsys	50
bode	63	feedback	23	pzmap	77
c2d, c2dt	34	freqs	74	reg, dreg	33
c2dm	35	freqz	74	rlocfind	79
canon	42	gram, dgram	45	rlocus	78
cloop	25	impulse	55	rmodel, drmodel	26
connect, blkbuild	28	initial	58	series	22
covar, dcovar	48	lqe, lqe2, lqew	82	sgrid	80
ctrb, obsv	45	lqed	84	sigma	72
ctrbf, obsvf	43	lqr, lqr2, lqry	84	ss2ss	44
d2c	37	lqrd	86	ss2tf	38
d2cm	37	lsim	60	ss2zp	38
damp, ddamp	47	ltifr	76	ssdelete	27
dbode	65	ltitr	63	ssselect	28
dcgain, ddcgain	46	lyap, lyap2, dlyap	52	step	53
dimpulse	57	margin	75	tf2ss	39
dinitial	60	minreal	41	tf2zp	39
dlqe, dlqew	83	modred, dmodred	42	tzero	50
dlqr, dlqry	85	ngrid	72	zgrid	80
dlsim	62	nichols	69	zp2ss	40
dnichols	71	nyquist	66	zp2tf	40
dsigma	73	ord2	26		

2.1 模型建立

1. augstate

功能：将状态增广到状态空间系统的输出中。

格式：

[ab，bb，cb，db]=augstate(a，b，c，d)

说明：

augstate 函数可将状态加到状态空间系统的输出中。[ab，bb，cb，db]=augstate(a,b,c,d)可产生一个新的状态空间系统，其输入和状态与原系统相同，但输出增加了所有的状态，即系统成为

$$\dot{x} = ax + bu$$

$$\begin{bmatrix} y \\ x \end{bmatrix} = \begin{bmatrix} c \\ I \end{bmatrix} x + \begin{bmatrix} d \\ 0 \end{bmatrix} u$$

这个命令是为 feedback 函数作准备，这样构成的系统可采用 feedback 命令构成全状态反馈的闭环系统。

参见：parallel，series，feedback，cloop

2. append

功能：两个状态空间系统的组合。

格式：

[a,b,c,d]=append(a1,b1,c1,d1,a2,b2,c2,d2)

说明：

append 函数可将两个状态空间系统进行组合，如图 2.1 所示。

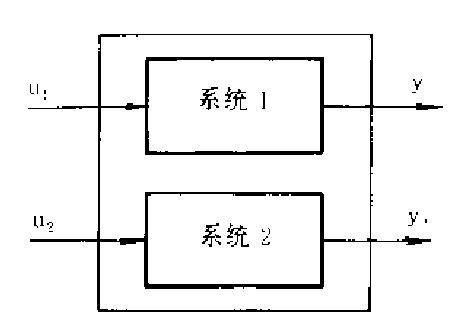


图 2.1　两系统的组合

[a,b,c,d]=append(a1,b1,c1,d1,a2,b2,c2,d2)，可将两系统按图 2.1 所示方式进行组合，得到的系统为

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}a_1 & 0\\ 0 & a_2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}b_1 & 0\\ 0 & b_2\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

$$\begin{bmatrix}y_1\\ y_2\end{bmatrix}=\begin{bmatrix}c_1 & 0\\ 0 & c_2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}d_1 & 0\\ 0 & d_2\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}$$

参见：augstate，parallel，series，cloop，feedback，are

3. parallel

功能：系统的并联连接。

格式：

```
[a,b,c,d]=parallel(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d]=parallel(a1,b1,c1,d1,a2,b2,c2,d2,inp1,inp2,out1,out2)
[num,den]=parallel(num1,den1,num2,den2)
```

说明：

parallel 函数按并联方式连接两个状态空间系统，它既适合于连续时间系统，也适合于离散时间系统。

[a,b,c,d]=parallel(a1,b1,c1,d1,a2,b2,c2,d2)，可得到由系统 1 和系统 2 并行联接的状态空间表示的系统，其输出为 $y=y_1+y_2$，其输入连接在一起，并作为系统输入，如图 2.2 所示。

由此得到的系统为

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}a_1 & 0\\ 0 & a_2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}b_1\\ b_2\end{bmatrix}u$$

$$y=y_1+y_2=[c_1\quad c_2]\begin{bmatrix}x_1\\ x_2\end{bmatrix}+[d_1\quad d_2]u$$

[num,den]=parallel(num1,den1,num2,den2)，可得到并联连接的传递函数表示系统，其结果为

$$\frac{num(s)}{den(s)}=g_1(s)+g_2(s)=\frac{num1(s)den2(s)+num2(s)den1(s)}{den1(s)den2(s)}$$

[a,b,c,d]=parallel(a1,b1,c1,d1,a2,b2,c2,d2,inp1,inp2,out1,out2)，可将系统 1 与系统 2 按图 2.3 所示的方式连接，在 inp1 和 inp2 中分别指定两系统要连接在一起的输入

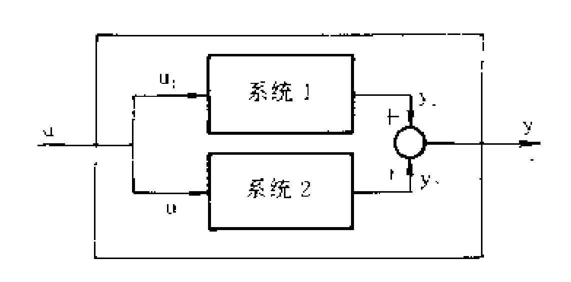


图 2.2　系统的并联连接

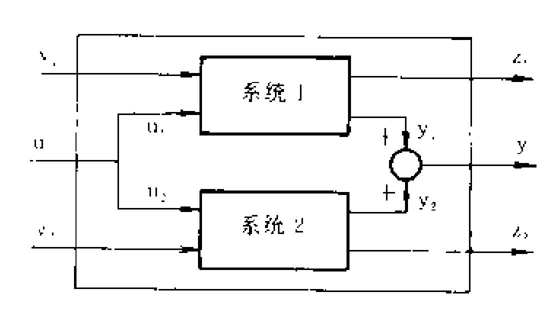


图 2.3　部分并联连接的系统

端编号，out1 和 out2 中分别指定要作相加的输出端编号。

例 两子系统为

$$g_1(s) = \frac{3}{s+4}$$

$$g_2(s) = \frac{2s+4}{s^2+2s+3}$$

将两者作并联连接，可输入

```
num1=3;
den1=[1, 4];
num2=[2, 4];
den2=[1, 2, 3];
[num,den]=parallel(num1,den1,num2,den2)
```

则得

```
num=
     0     5     18     25
den=
     1     6     11     12
```

因此

$$g(s) = g_1(s) + g_2(s) = \frac{5s^2+18s+25}{s^3+6s^2+11s+12}$$

参见：append，cloop，feedback，series

4. series

功能：系统的串联连接。

格式：

```
[a,b,c,d]=series(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d]=series(a1,b1,c1,d1,a2,b2,c2,d2,outputs1,inputs2)
[num,den]=series(num1,den1,num2,den2)
```

说明：

series 函数可将两个系统按串联方式进行连接，它既适用于连续时间系统，也适用于离散时间系数。

[a,b,c,d]=series(a1,b1,c1,d1,a2,b2,c2,d2)，可将系统 1 的所有输出连到系统 2 的输入上，即 $u_2=y_1$，这样得到了如图 2.4 所示的系统

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} a_1 & 0 \\ b_2c_1 & a_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2d_1 \end{bmatrix} u_1$$

$$y_2 = \begin{bmatrix} d_2c_1 & c_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [d_2d_1]\, u_1$$

[num,den]=series(num1,den1,num2,den2)可得到串联连接的传递函数形式

$$\frac{num(s)}{den(s)} = g_1(s)g_2(s) = \frac{num1(s)num2(s)}{den1(s)den2(s)}$$

[a,b,c,d]=series(a1,b1,c1,d1,a2,b2,c2,d2,outputs1,inputs2)，可将系统 1 和系统 2 按图 2.5 所示的方式进行连接，outputs1 和 inputs2 用于指定系统 1 的部分输出和系统 2 的部分输入进行连接。

参见：append，cloop，feedback，parallel，connect

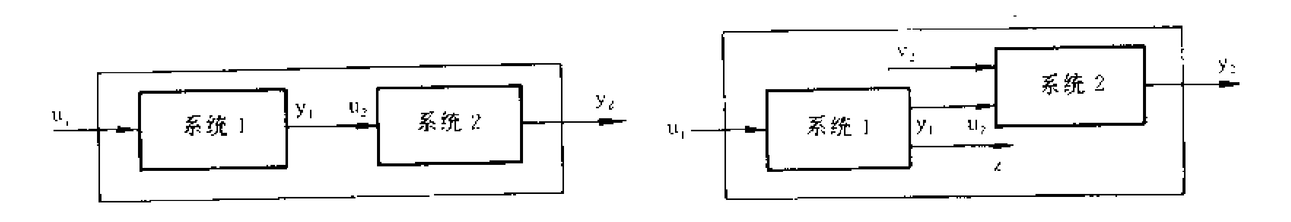


图 2.4　系统的串联连接　　　　图 2.5　部分串联连接的系统

5. feedback

功能：两个系统的反馈连接。

格式：

```
[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2,sign)
[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2,inp1,out1)
[num,den]=feedback(num1,den1,num2,den2)
[num,den]=feedback(num1,den1,num2,den2,sign)
```

说明：

feedback 函数可将两个系统按反馈形式进行连接，如图 2.6 所示。一般而言，系统 1 为对象，系统 2 为反馈控制器。feedback 函数既适用于连续系统，也适用于离散系统。

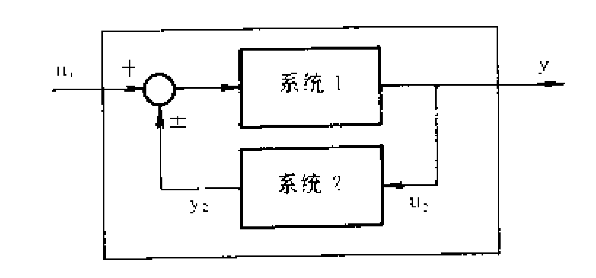


图 2.6　两系统的反馈连接

[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2,sign)，可将两系统按反馈方式连接起来，系统 1 的所有输出连接到系统 2 的输入，系统 2 的所有输出连接到系统 1 的输入，sign 是用于指示 y_2 到 u_1 连接的符号，sign 缺省时，默认为负，即 sign=−1。总系统的输入/输出数等同于系统 1。

连接产生的系统为

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}a_1 + b_1ed_2c_1 & \pm b_1ec_2\\ b_2c_1 \pm b_2d_1ed_2c_1 & a_2 \pm b_2d_1ec_2\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}b_1(I \pm ed_2d_1)\\ b_2d_1(I \pm ed_2d_1)\end{bmatrix}u_1$$

$$y_1 = [c_1 \pm d_1ed_2c_1 \quad \pm d_1ec_2]\begin{bmatrix}x_1\\ x_2\end{bmatrix} + [d_1(I \pm ed_2d_1)]u_1$$

其中

$$e = (I \mp d_2 d_1)^{-1}$$

式中下面的符号对应于正反馈，上面的符号对应于负反馈。

[num,den]=feedback(num1,den1,num2,den2,sign)可得到类似的连接，只是子系统和闭环系统均以传递函数形式表示。sign 的含义与上述相同。若以 g(s)=num(s)/den(s)表示，则最后由 feedback 函数得到的系统可表示成

$$\frac{num(s)}{den(s)} = \frac{g_1(s)}{1 \mp g_1(s)g_2(s)} = \frac{num1(s)den2(s)}{den1(s)den2(s) \mp num1(s)num2(s)}$$

为将两个均出现在前向通道中的系统传递函数按反馈方式连接起来，应采用 series 和 cloop 函数。

[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2,inp1,out1)将系统 1 的指定输出(out1)连接到系统 2 的输入，系统 2 的输出连接到系统 1 的指定输入(inp1)，以此构成闭环系统，如图 2.7 所示。

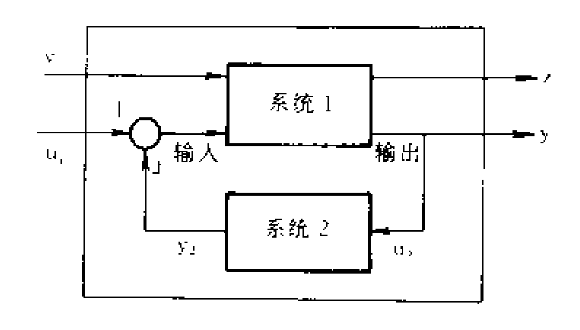


图 2.7　部分反馈连接

例　有两个子系统

$$g(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

$$h(s) = \frac{5(s + 2)}{s + 10}$$

现按图 2.6 方式连接，求闭环系统的传递函数。

利用 feedback 函数可很容易求得，在 MATLAB 中输入

```
numg=[2  5  1];
deng=[1  2  3];
numh=[5  10];
denh=[1  10];
[num,den]=feedback(numg,deng,numh,denh)
```

执行后得

```
num=
       2     25     51     10
den=
      11     57     78     40
```

因此闭环系统的传递函数为

$$g_c(s) = \frac{num(s)}{den(s)} = \frac{2s^3 + 25s^2 + 51s + 10}{11s^3 + 57s^2 + 78s + 40}$$

参见：append，cloop，connect，parallel，series

6. cloop

功能：状态空间系统的闭环形式。

格式：

[ac,bc,cc,dc]=cloop(a,b,c,d,sign)

[ac,bc,cc,dc]=cloop(a,b,c,d,outputs,inputs)

[numc,denc]=cloop(num,den,sign)

说明：

cloop 函数可通过将系统输出反馈到系统输入构成闭环系统。开环系统的输入/输出仍然是闭环系统的输入/输出。cloop 函数既适用于连续系统，也适用于离散系统。

[ac,bc,cc,dc]=cloop(a,b,c,d,sign)，通过将所有的输出反馈到输入，从而产生闭环系统的状态空间模型，如图 2.8 所示。

当 sign=1 时采用正反馈；当 sign=－1 时采用负反馈；sign 缺省时，默认为负反馈。

由 cloop 函数产生的正、负反馈的闭环系统为

$$\dot{x} = [a \pm b(I \mp d)^{-1}c]x + [b(I \mp d)^{-1}]u$$

$$y = [c \pm d(I \mp d)^{-1}c]x + [d(I \mp d)^{-1}]u$$

[numc,denc]=cloop(num,den,sign)，表示由传递函数表示的开环系统构成闭环系统，sign 意义与上述相同，由此得到正、负反馈闭环系统为

$$\frac{numc(s)}{denc(s)} = \frac{g(s)}{1 \mp g(s)} = \frac{num(s)}{den(s) \mp num(s)}$$

[ac,bc,cc,dc]=cloop(a,b,c,d,outputs,inputs)，表示将指定的输出 outputs 反馈到指定的输入 inputs，以此构成闭环系统的状态空间模型，如图 2.9 所示。其中 outputs 指定反馈系统输出序号，inputs 指定输入序号。一般为正反馈，形成负反馈时应在 inputs 中采用负值。

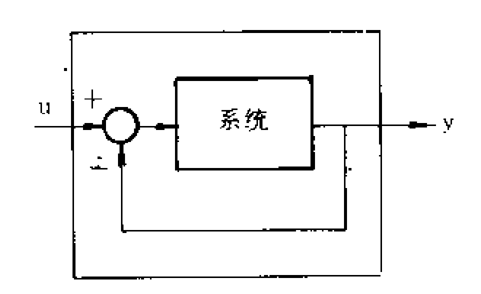


图 2.8　系统的闭环

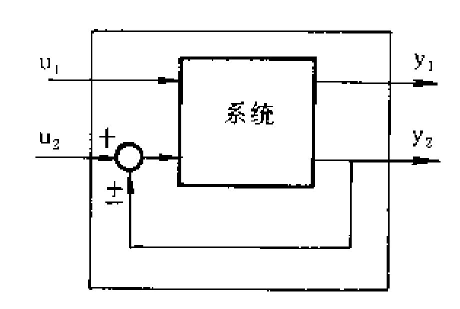


图 2.9　系统部分反馈的闭环

例　状态空间系统(a,b,c,d)具有 8 输入 5 输出，现将第 1、3、5 个输出负反馈到第 2、8、7 输入中，以此构成闭环系统，可输入

```
outputs=[1  3  5];
inputs=[-2  -8  -7];
[ac,bc,cc,dc]=cloop(a,b,c,d,outputs,inputs);
```

参见：append，connect，feedback，parallel，series

7. ord2

功能：产生二阶系统。

格式：

```
[a,b,c,d]=ord2(Wn,z)
[num,den]=ord2(Wn,z)
```

说明：

[a,b,c,d]=ord2(Wn,z)可得到二阶系统

$$h(s) = \frac{1}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

的状态空间表示(a,b,c,d)，其中 z 表示 ξ，Wn 表示 ω_n。

[num,den]=ord2(Wn,z)可得到二阶系统的传递函数表示。

例 要产生 $\xi=0.4$，$\omega_n=2.4$ 弧度/秒的二阶系统的传递函数，可输入

```
[num,den]=ord2(2.4,0.4)
```

则得

```
num=
      1
den=
      1      1.9200      5.7600
```

因此有

$$h(s) = \frac{1}{s^2 + 1.92s + 5.76}$$

参见：blkbuild，connect，tf2ss，ss2tf

8. rmodel，drmodel

功能：稳定的随机 n 阶模型。

格式：

```
[a,b,c,d]=rmodel(n)
[a,b,c,d]=rmodel(n,p,m)
[num,den]=rmodel(n)
[num,den]=rmodel(n,p)

[a,b,c,d]=drmodel(n)
[a,b,c,d]=drmodel(n,p,m)
[num,den]=drmodel(n)
[num,den]=drmodel(n,p)
```

说明：

[a,b,c,d]=rmodel(n)可得到随机的 n 阶稳定状态空间模型(a,b,c,d)，并且为 SISO

系统。

[a,b,c,d]=rmodel(n,p,m)可得到一个 m 输入 p 输出的随机 n 阶稳定模型。

[num,den]=rmodel(n)可得到随机的 n 阶稳定传递函数模型。

[num,den]=rmodel(n,p)可得到一个单输入 p 输出的 n 阶随机稳定模型。

drmodel 可得到稳定的离散时间随机模型。

参见：无

9. ssdelete

功能：从状态空间系统中删除输入、输出或状态。

格式：

[ar,br,cr,dr]=ssdelete(a,b,c,d,inputs,outputs)

[ar,br,cr,dr]=ssdelete(a,b,c,d,inputs,outputs,states)

说明：

给定状态空间系统

$$\dot{x} = ax + \begin{bmatrix} b_1 & b_2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} x + \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[ar,br,cr,dr]=ssdelete(a,b,c,d,inputs,outputs)，可从状态空间系统(a,b,c,d)中删除指定的输入和输出，inputs 用于指定要删除的输入编号，outputs 用于指定要删除的输出编号。

例 在上述系统中要删除输入 u_2 和输出 y_2，则得到的状态空间系统为

$$\dot{x} = ax + b_1u_1$$

$$y_1 = c_1x + d_{11}u_1$$

ssdelete 可以用来删除系统中的慢变状态，其结果等效于将这些删除状态设定为常值或 0 时的简化系统。

给定系统

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[ar,br,cr,dr]=ssdelete(a,b,c,d,inputs,outputs,states)，可从状态空间系统中删除指定的输入、输出和状态。

例 要删除输入 u_2、输出 y_2 和状态 x_2，则得到的状态空间系统为

$$\dot{x} = a_{11}x_1 + b_1u_1$$

$$y_1 = c_{11}x_1 + d_{11}u_1$$

ssdelete 函数既适用于连续系统，也适用于离散系统。

参见：ssselect

10. ssselect

功能：从大状态空间系统中选择一个子系统。

格式：

```
[ae,be,ce,de]=ssselect(a,b,c,d,inputs,outputs)
[ae,be,ce,de]=ssselect(a,b,c,d,inputs,outputs,states)
```

说明：

给定状态空间系统

$$\dot{x} = ax + [b_1 \quad b_2]\begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} x + \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}$$

[ae,be,ce,de]=ssselect(a,b,c,d,inputs,outputs)，可利用指定的输入和输出产生一个子系统，inputs 用于指定子系统的输入，ouputs 用于指定子系统的输出。

在上述状态系统中，设 u_1 选作子系统的输入、y_1 选作子系统的输出，则得到的状态空间系统为

$$\dot{x} = ax + b_1 u_1$$

$$y_1 = c_1 x + d_{11} u_1$$

[ae,be,ce,de]=ssselect(a,b,c,d,inputs,outputs,states)，可利用指定的输入、输出及状态构造子系统，其输入、输出和状态由 inputs、outputs、states 指定。

ssselect 既适用于连续系统，也适用于离散系统。

例 考虑一个具有 5 输出 4 输入的状态空间系统(a,b,c,d)，若以输入 1 和 2，输出 2、3、4 构成一个子系统，则可输入

```
inputs=[1  2];
outputs=[2  3  4];
[ae,be,ce,de]=ssselect(a,b,c,d,inputs,outputs);
```

参见：ssdelete

11. connect，blkbuild

功能：将方框图转换为状态空间模型。

格式：

```
blkbuild
[ac,bc,cc,dc]=connect(a,b,c,d,q,inputs,outputs)
```

说明：

[ac,bc,cc,dc]=connect(a,b,c,d,q,inputs,outputs)可得到状态空间模型(ac,bc,cc,dc)，其中 a、b、c、d 是给定的无连接对角方块，q 用于指定内部连接关系，inputs 和 outputs 用于选择系统(ac,bc,cc,dc)的输入和输出。

首先，看看函数 connect 的建模过程，以 SISO 传递函数构成的方框图为例。

(1) 定义传递函数或状态空间系统

在方框图上对每个方框进行编号，并输入相应的传递函数分子分母系数：n1,n2,n3,…和 d1,d2,d3,…，或者输入状态矩阵(a,b,c,d)：a1,b1,c1,d1；a2,b2,c2,d2；a3,b3,c3,d3；…。

(2) 建立无连接的状态空间模型

形成一个由所有无连接关系传递函数构成的对角块模型(a,b,c,d)，这可以通过重复调用 append 或 tf2ss 和 append 来完成。例如，对以传递函数表示的系统

```
[a,b,c,d]=tf2ss(n1,d1);
[at,bt,ct,dt]=tf2ss(n2,d2);
[a,b,c,d]=append(a,b,c,d,at,bt,ct,dt);
[at,bt,ct,dt]=tf2ss(n3,d3);
[a,b,c,d]=append(a,b,c,d,at,bt,ct,dt);
```

或者对以状态空间表示的系统

```
[a,b,c,d]=append(a1,b1,c1,d1,a2,b2,c2,d2);
[a,b,c,d]=append(a,b,c,d,a3,b3,c3,d3);
```

另外，还可使用一个称作为 blkbuild 的函数，它可自动完成上述过程。例如

```
% Define either transfer functions or state space models
...
nblocks=5;
blkbuild
```

可完成由五个方框构成的系统。

blkbuild 能自动确定每个方框是状态空间形式还是传递函数形式，在形成的系统中包含状态空间方框的所有输入和输出，输入/输出顺序与方框顺序相对应。

(3) 指定方框间的连接关系

矩阵 q 用于指示方框间的连接关系，矩阵的每一行对应于一个输入，其第一个元素为输入编号，其后为连接该输入的输出编号，如采用负连接，则以负值表示。例如，输入 7 由输出 2、15、6 构成，其中输出 15 至输入 7 之间为负连接，则对应的 q 矩阵中这一行为[7　2　−15　6]。

(4) 选择输入/输出

inputs 和 outputs 用于指示无连接系统中的某些输入/输出保留作外部的输入/输出，例如

```
inputs=[1  2  15];
outputs=[2  7]
```

则表示无连接系统中的输入 1、2、15 用作系统输入，输出 2、7 用作系统的输出。

(5) 内部连接

调用[ac,bc,cc,dc]=connect(a,b,c,d,q,inputs,outputs)这一函数，可以从矩阵 q 中获得连接信息，对方框图(a,b,c,d)进行连接，从而得到系统(ac,bc,cc,dc)，其输入和输出分别由 inputs 和 outputs 确定。

为简化上述过程，建议生成 MATLAB 程序文件，这样便于修改和检查。在构成新的

系统后，还应该作进一步的检查。

例 某系统包含有一个 MIMO 块，其方框图如图 2.10 所示。

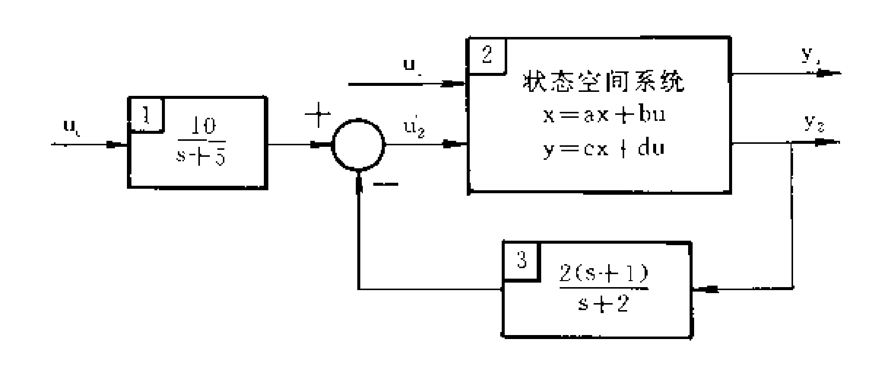


图 2.10 MIMO 系统

在 MATLAB 中，得到系统的状态空间模型的 M 程序如下

```
% Feedforward transfer function
n1=10;
d1=[1  5];
% state-space plant
a2=[-9.0201   17.7791
     -1.6943    3.2138];
b2=[-.5112       .5362
     -.0020    -1.8470];
c2=[-3.2897      2.4544
     -13.5009  18.0745];
d2=[-.5476   -.1410
     -.6459     .2958];
% Transfer function controller
n3=2*[1  1];
d3=[1  2];
nblocks=3;
blkbuild;
```

执行这一 MATLAB 程序，可得

```
a=
   -5.0000       0          0          0
      0      -9.0201    17.7791        0
      0      -1.6943     3.2138        0
      0          0          0      -2.0000
b=
   1.0000        0          0          0
      0      -0.5112     0.5362        0
      0      -0.0020    -1.8470        0
      0          0          0      1.0000
```

```
c=
   10.0000         0          0          0
        0    -3.2897     2.4544          0
        0   -13.5009    18.0745          0
        0          0          0    -2.0000
d=
   0         0         0         0
   0   -0.5476   -0.1410         0
   0   -0.6459   -0.2958         0
   0         0         0    2.0000
```

再输入

```
q=[3  1  -4
   4  3   0];
inputs=[1  2];
outputs=[2  3];
[ac,bc,cc,dc]=connect(a,b,c,d,q,inputs,outputs)
```

执行后得到闭环系统

```
ac=
    -5.0000          0          0          0
     3.3689     0.0766     5.6007     0.6738
   -11.6047   -33.0290    45.1635    -2.3209
     1.8585    -8.4826    11.3562    -1.6283
bc=
    1.0000          0
         0    -0.0760
         0    -1.5011
         0    -0.4058
cc=
   -0.8859   -5.6818    5.6568   -0.1772
    1.8585   -8.4826   11.3562    0.3717
dc=
   0   -0.6620
   0   -0.4058
```

参见：cloop，series，parallel，feedback，append，minreal

12. estim，destim

功能：生成连续/离散状态估计器或观测器。

格式：

```
[ae,be,ce,de]=estim(a,b,c,d,l)
```

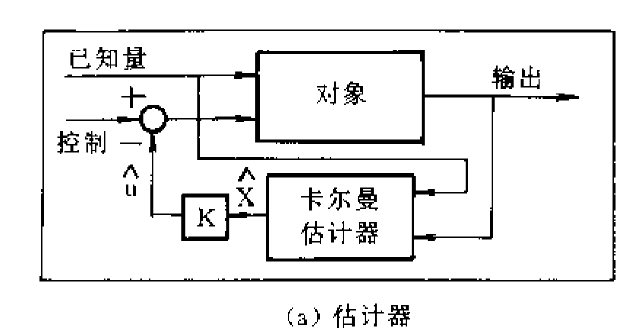
[ae,be,ce,de]=estim(a,b,c,d,l,sensors,known)

[ae,be,ce,de]=destim(a,b,c,d,l)

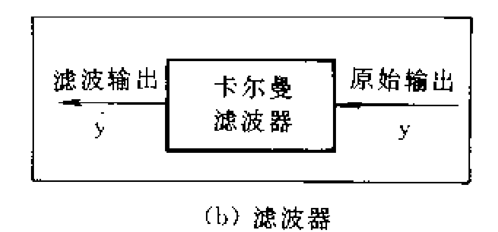
[ae,be,ce,de]=destim(a,b,c,d,l,sensors,known)

说明：

estim 和 destim 可从状态空间系统和增益矩阵 l 中生成稳态卡尔曼估计器。增益矩阵 l 通常由 lqe 或 dlqe 设计产生，也可以由极点配置函数 place 形成。在反馈控制中，卡尔曼估计器可用作状态估计器或滤波器，如图 2.11 所示。



(a) 估计器



(b) 滤波器

图 2.11　卡尔曼估计器

[ae,be,ce,de]=estim(a,b,c,d,l)可得到连续系统的状态估计器

$$\dot{x} = ax + bu$$

$$y = cx + du$$

假定系统的所有输出是传感器输出，则得到的状态估计器为

$$\dot{\hat{x}} = [a - lc]\hat{x} + ly$$

$$\begin{bmatrix}\hat{y}\\ \hat{x}\end{bmatrix} = \begin{bmatrix}c\\ I\end{bmatrix}\hat{x}$$

[ae,be,ce,de]=estim(a,b,c,d,sensors,known)可得到连续状态的估计器，其传感器由 sensors 指定，已知输入由 known 指定。由 known 指定的输入自动包括在估计器的输入中。

[ae,be,ce,de]=destim(a,b,c,d,l)可得到离散系统

$$\dot{x}[n+1] = ax[n] + bu[n]$$

$$y[n] = cx[n] + du[n]$$

的状态估计器，并假定系统的所有输出为传感器的输出，这时得到的状态估计器为

$$\bar{x}[n+1] = [a - alc]\bar{x}[n] + aly[n]$$

$$\begin{bmatrix}\hat{y}[n]\\ \hat{x}[n]\end{bmatrix} = \begin{bmatrix}c - clc\\ I - lc\end{bmatrix}\bar{x}[n] + \begin{bmatrix}cl\\ l\end{bmatrix}y[n]$$

[ae,be,ce,de]=destim(a,b,c,d,l,sensors,known)将采用指定的传感器输出 sensors 和指定的已知输入 known 产生状态估计器。

例　考虑一个由 7 输出和 4 输入构成的状态空间系统(a,b,c,d)，当系统的输出 4、7 和 1 作为传感器输出，输入 1、4 和 3 作为已知输入时，设计卡尔曼增益矩阵 l，从而得到状态估计器，这时 MATLAB 程序为

```
sensors=[4,7,1];
```

```
known=[1,4,3];
[ae,be,ce,de]=estim(a,b,c,d,l,sensors,known);
```

参见：reg，dreg，lqe，dlqe，dlqr

13. reg，dreg

功能：形成控制器/估计器。

格式：

```
[ac,bc,cc,dc]=reg(a,b,c,d,k,l)
[ac,bc,cc,dc]=reg(a,b,c,d,k,l,sensors,known,controls)
[ac,bc,cc,dc]=dreg(a,b,c,d,k,l)
[ac,bc,cc,dc]=dreg(a,b,c,d,k,l,sensors,known,controls)
```

说明：

reg 和 dreg 可以从状态空间系统、反馈增益矩阵 k 及估计器增益矩阵 l 中形成控制器/估计器。增益矩阵通常可利用线性二次高斯法 lqe、lqr 或 dlqe、dlqr 进行设计。对利用 feedback 构成的闭环系统来说，构造控制器/估计器是很有必要的。

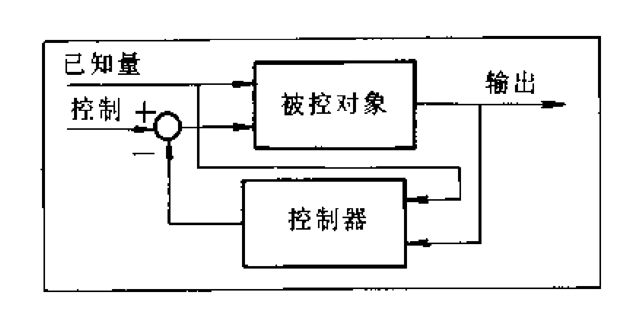


图 2.12 控制器与对象的连接

由 reg 和 dreg 得到控制器与对象之间的连接，如图 2.12 所示。

[ac,bc,cc,dc]=reg(a,b,c,d,k,l)可生成连续系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

的控制器/估计器，并假定被控对象的所有输入为控制输入，对象的所有输出为传感器输出。这样得到的控制器/估计器为

$$\dot{\hat{x}} = [a - bk - lc + ldk]\,\hat{x} + ly$$
$$\hat{u} = k\hat{x}$$

[ac,bc,cc,dc]=reg(a,b,c,d,k,l,sensors,known,controls)，sensors 用于指定传感器输出，knowns 用于指定已知输入，controls 用于指定控制输入，以此构成控制器/估计器。

[ac,bc,cc,dc]=dreg(a,b,c,d,k,l)可生成离散系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

的控制器/估计器，并假定对象的所有输入为控制输入，对象的所有输出为传感器输出。这样得到的控制器/估计器为

$$\bar{x}[n+1] = [a - alc - (b - ald)e(k - klc)]\,\bar{x}[n] + [al - (b - ald)ekl]\,y[n]$$
$$\hat{u}[n] = [k - klc + klde(k - klc)]\,\bar{x}[n] + [kl + kldekl]\,y[n]$$

$$e = (I - kld)^{-1}$$

[ac,bc,cc,dc]=dreg(a,b,c,d,k,l,sensors,known,controls)，利用指定的传感器输出、已知输入、控制输入构造离散控制器/估计器。

例 考虑一个由 7 输出/4 输入的连续状态空间系统(a,b,c,d)，利用受控对象的第 1、2、4 个输入作为控制输入构成增益矩阵 k，利用第 4、7、1 个输出作为传感器输出构成 kalman增益矩阵 l，输入 3 构成已知输入，这样构成控制器/估计器。可输入

```
controls=[1,2,4];
sensors=[4,7,1];
known=[3];
[ac,bc,cc,dc]=reg(a,b,c,d,k,l,sensors,known,controls);
```

参见：estim，destim，lqe，lqr，dlqe，dlqr，cloop，feedback

14. pade

功能：时延的 pade 近似。

格式：

```
[a,b,c,d]=pade(t,n)
[num,den]=pade(t,n)
```

说明：

pade 函数可产生时延环节的 n 阶 LTI 逼近模型。这种 pade 近似模型可用来对连续系统的传输和计算等延时进行建模，t 秒时延的拉氏变换为 e^{-st}，它可由 pade 近似

$$e^{-st} = 1 - st + \frac{1}{2!}(st)^2 - \frac{1}{3!}(st)^3 + \cdots \approx \frac{num(s)}{den(s)}$$

当 num(s)和 den(s)为 n 阶时，pade 近似可使截断误差最小。

[a,b,c,d]=pade(t,n)可产生最佳逼近时延 t 秒的 n 阶 SISO 的状态空间模型。

[num,den]=pade(t,n)产生最佳逼近时延 t 秒的 n 阶传递函数形式。

例 求时延 0.2 秒的 5 阶 pade 近似，可输入

```
[num,den]=pade(0.2, 5);
```

参见：c2d，c2dt

2.2 模型变换

1. c2d，c2dt

功能：变连续时间系统为离散时间系统。

格式：

```
[ad,bd]=c2d(a,b,Ts)
[ad,bd,cd,dd]=c2dt(a,b,c,Ts,lambda)
```

说明：

c2d 和 c2dt 完成将状态空间模型从连续时间到离散时间的转换，并且假定系统输入采用零阶保持器，c2dt 函数的输入中还可包含时间延迟 lambda。

[ad,bd]=c2d(a,b,Ts)可将连续时间状态空间系统

$$\dot{x} = ax + bu$$

变换成离散时间系统

$$x[n+1] = a_d x[n] + b_d u[n]$$

并且假定控制输入为 Ts 上的分段常数。

[ad,bd,cd,dd]=c2dt(a,b,c,Ts,lambda)，可将具有输入纯延时 λ(用 lambda 表示)的连续时间状态空间系统

$$\dot{x}(t) = ax(t) + bu(t-\lambda)$$
$$y(t) = cx(t)$$

转换成离散时间系统

$$x[n+1] = a_d x[n] + b_d u[n]$$
$$y[n] = c_d x[n] + d_d u[n]$$

其中 Ts 为取样时间，λ 为输入延时，且满足

$$-Ts < \lambda < \infty$$

λ 取负值表示预测而不是延迟，这样可得到取样常数之间的系统响应。

c2d 与 d2c 互为逆操作。

参见：d2c，c2dm，expm，logm，funm

2．c2dm

功能：变连续状态空间模型为离散状态空间模型。

格式：

[ad,bd,cd,dd]=c2dm(a,b,c,d,Ts,'method')

[numd,dend]=c2dm(num,den,Ts,'method')

说明：

c2dm 可实现状态空间模型从连续时间域变换成离散时间域，变换方法由 method 指定：

'zoh' 在变换中，输入端采用零阶保持器，即在采样时间 Ts 上假定控制输入为分段常数(这与 c2d 相同)

'foh' 在变换中，输入端采用一阶保持器，即在采样时间 Ts 上假定控制输入为分段线性，这种变换是不可逆的，也就是说，不存在 d2cm(…,'foh')

'tustin' 在变换中，采用双线性(tustin)逼近导数

'prewarp' 在变换中，利用频率预变的双线性(tustin)来进行逼近

'matched' 利用匹配零—极点方法将 SISO 系统变换成离散时间系统

如果缺省 method，则默认为'zoh'。

[ad,bd,cd,dd]=c2dm(a,b,c,d,Ts,'method')，采用'method'指定的方法，将连续时

间状态空间系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

变换成离散时间系统

$$x[n+1] = a_d x[n] + b_d u[n]$$
$$y[n] = c_d x[n] + d_d u[n]$$

[numd,dend]=c2dm(num,den,Ts,'method')是以传递函数形式表示的从连续时间到离散时间的变换

```
c2dm(a,b,c,d,Ts,'method')
c2dm(num,den,Ts,'method')
```

命令可在屏幕上绘出对比的奇异值或波特增益图。连续系统的响应用实线表示，离散系统响应用虚线表示。

双线性近似(或 tustin 方法)可以将离散系统由 ω 域转换到 z 域，在这种情况下,d2cm(…,'tustin')将离散系统 z 域转换到 ω 域，而 c2dm 将 ω 域转换到 z 域。

c2dm 和 d2cm 与 expm 和 logm 类似，只要不超过奈奎斯特速率，它们就是互逆操作。

例 用 tustin 方法来离散化两输入三输出的状态空间系统，并比较两者的奇异值响应。可输入

```
[a1,b1,c1,d1]=ord2(1,0.2);
[a2,b2,c2,d2]=ord2(0.7,0.35);
[a,b,c,d]=series(a1,b1,c1,d1,a2,b2,c2,d2);
Ts=1;
c2dm(a,b,c,d,Ts,'tustin');
title('Continuous/Discrete Singular Value Comparison')
```

执行后得到如图 2.13 所示的曲线。

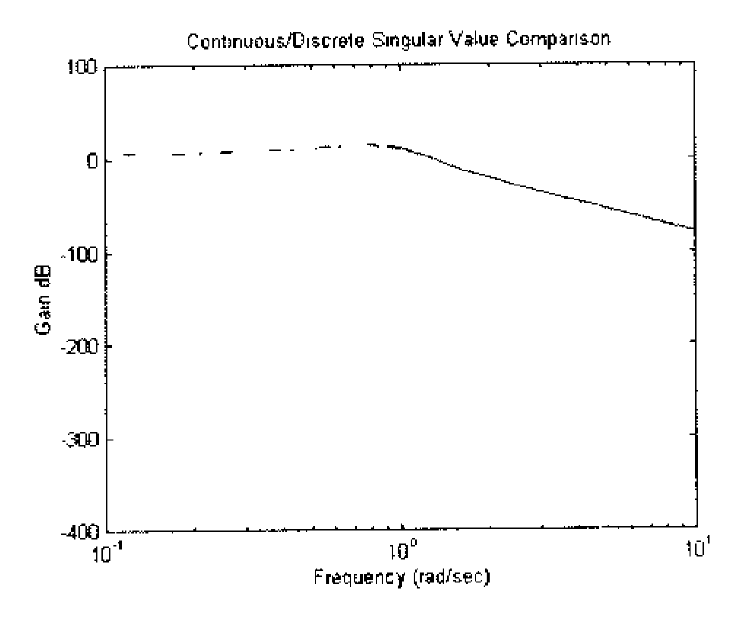


图 2.13 奇异值比较曲线

参见：expm，c2d，d2cm，logm，funm

3．d2c

功能：变离散时间系统为连续时间系统。

格式：

```
[ac,bc]=d2c(a,b,Ts)
```

说明：

d2c 函数可将状态空间模型从离散时间变换到连续时间，并且假定输入采用零阶保持器。

[ac,bc]=d2c(a,b,Ts)可将离散时间状态空间系统

$$x[n+1]=ax[n]+bu[n]$$

变换成连续时间系统

$$\dot{x}=a_c x+b_c u$$

并且假定在取样时间 Ts 上控制输入为分段常数。d2c 和 c2d 互为逆操作。

参见：expm，c2d，c2dm，logm，funm

4．d2cm

功能：变离散时间系统为连续时间系统。

格式：

```
[ac,bc,cc,dc]=d2cm(a,b,c,d,Ts,'method')
[numc,denc]=d2cm(num,den,Ts,'method')
```

说明：

d2cm 函数可采用指定的方法完成从离散时间到连续时间系统的变换，变换方法由 method 指定，可取'zoh'，'tustin'，'prewarp'，'matched'，详细说明可参见 c2dm 函数。

d2cm 是 c2dm 的逆过程，因此有关规定或说明可参见 c2dm 函数。

例　用 tustin 方法将两输入三输出的离散状态空间系统变换成连续时间系统，并对奇异值响应进行比较。可输入

```
a=[0,1; -1,-0.4];
b=[0; 1];
c=[1,0];
d=[0];
Ts=1;
% plot comparison graph
d2cm(a,b,c,d,Ts,'tustin')
titlel ('Continues/Discrete Singular Value Comparison')
```

执行后可得如图 2.14 所示的曲线。

参见：expm，c2dm，d2c，logm，funm

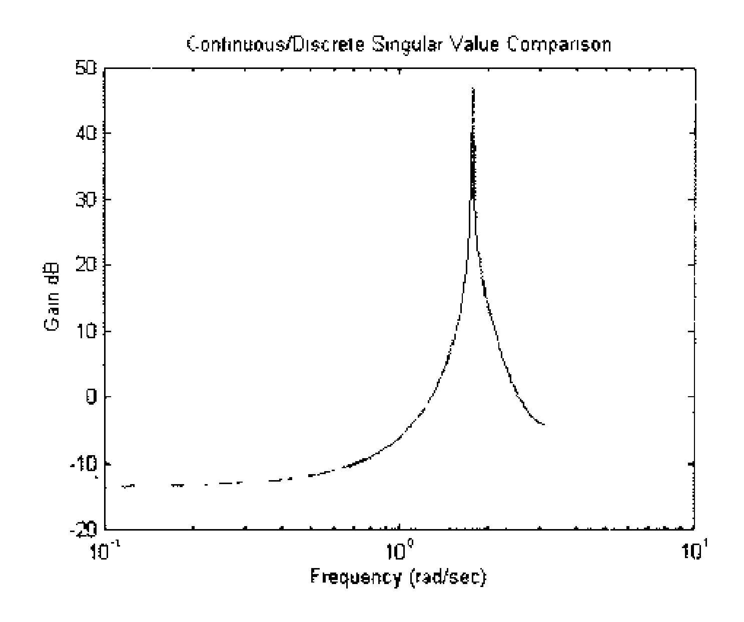


图 2.14　奇异值比较曲线

5. ss2tf

功能：变系统状态空间形式为传递函数形式。

格式：

[num, den]=ss2tf(A, B, C, D, iu)

说明：

系统的状态方程可表示为

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

相应的传递函数为

$$H(s) = \frac{num(s)}{den(s)} = C(sI - A)^{-1}B + D$$

[num, den]=ss2tf(A, B, C, D, iu)可将状态空间表示变换成相应的传递函数表示，iu 用于指定变换所使用的输入量。

ss2tf 函数还可以应用于离散时间系统，这时得到的是 Z 变换表示。

参见：ss2zp，tf2ss，tf2zp，zp2ss，zp2tf

6. ss2zp

功能：变系统状态空间形式为零极点增益形式。

格式：

[z, p, k]=ss2zp(A, B, C, D, iu)

说明：

系统状态空间表示和零极点增益表示可分别参考 sos2ss 和 sos2zp 函数(详见《基于 MATLAB 的系统分析与设计——信号处理》)。[z, p, k]=ss2zp(A, B, C, D, iu)可将状

态空间表示转换成零极点增益表示，iu 用于指定变换所用的输入量。

ss2zp 函数还可以应用于离散时间系统，这时得到的是 Z 变换表示。

参见：ss2tf，tf2ss，zp2ss

7．tf2ss

功能：变系统传递函数形式为状态空间形式。

格式：

[A，B，C，D]=tf2ss(num，den)

说明：

tf2ss 函数可将给定系统的传递函数表示变换成等效的状态空间表示。在[A,B,C,D]=tf2ss(num，den)格式中，矢量 den 按 s 的降幂顺序输入分母系数，矩阵 num 每一行为相应于某输出的分子系数，其行数为输出的个数。tf2ss 得到控制器正则形式的 A、B、C、D 矩阵。

tf2ss 也可以用于离散系统中，但这时必须在分子多项式中补零，以使分子分母的长度相同。

例 将系统

$$H(s)=\frac{\begin{bmatrix}2s+3\\ s^2+2s+1\end{bmatrix}}{s^2+0.4s+1}$$

变换成状态空间表示

```
num=[0 2 3; 1 2 1];
den=[1 0.4 1];
[A, B, C, D]=tf2ss(num, den)
A=
    -0.4000    -1.0000
     1.0000          0
B=
     1
     0
C=
   2.0000    3.0000
   1.6000         0
D=
     0
     1
```

参见：ss2tf，ss2zp，zp2ss

8．tf2zp

功能：变系统传递函数形式为零极点增益形式。

格式：

[z, p, k]=tf2zp(num, den)

说明：

tf2zp 函数可找出多项式传递函数形式的系统的零点、极点和增益。

tf2zp 函数类似于 ss2zp 函数。

例 要找出系统

$$H(s) = \frac{s^2 - 0.5s + 2}{s^2 + 0.4s + 1}$$

的零点、极点和增益，可以输入

```
num=[1 -0.5 2];
den=[1 0.4 1];
[z, p, k]=tf2zp(num, den)
z=
   0.2500+1.3919i
   0.2500-1.3919i
p=
   -0.2000-0.9798i
   -0.2000-0.9798i
k=
     1
```

参见：ss2tf, ss2zp, tf2ss, zp2tf

9. zp2ss

功能：变系统零极点增益形式为状态空间形式。

格式：

[A, B, C, D]=zp2ss(z, p, k)

说明：

[A,B,C,D]=zp2ss(z, p, k)可将以 z, p, k 表示的零极点增益形式变换成状态空间形式。

参见：ss2tf, ss2zp, tf2ss

10. zp2tf

功能：变系统零极点增益形式为传递函数形式。

格式：

[num, den]=zp2tf(z, p, k)

说明：

[num,den]=zp2tf(z,p,k)可将以 z,p,k 表示的零极点增益形式变换成传递函数形式。

参见：ss2tf, ss2zp, tf2ss, tf2zp, zp2ss

2.3 模型简化

1. balreal, dbalreal

功能：平衡状态空间的实现。

格式：

```
[ab,bb,cb,g,T]=balreal(a,b,c)
[ab,bb,cb,g,T]=dbalreal(a,b,c)
```

说明：

状态空间模型的一个重要实现就是所谓平衡实现。函数 balreal 用于查找具有相等但对角可控性和可观性的实现算法，这种平衡模型有利于模型的降阶。

[ab,bb,cb,g,T]=balreal(a,b,c)可得到系统(a,b,c)的平衡化参数和矢量 g 及矩阵 T，g 包含平衡实现的对角阵，T 类似于将(a,b,c)变换成(ab,bb,cb)。应该注意，当 ss2ss 函数中使用 T 时，应将 T 求逆。

例 可对一四阶系统进行平衡化处理

```
[ab,bb,cb,g,T]=balreal(a,b,c);
```

假如得到的 g 为

```
g=
   0.6342    0.4313    0.0812    0.0203
```

由此可见，可以清除后两个状态

```
[ar,br,cr]=modred(ab,bb,cb,[3,4]);
```

对归一化后的系统，矢量 g 可用于降阶。由于 g 中反映出了各状态的组合可控性和可观性，因此可从模型中删去 g 值小的状态，从而达到模型降阶的目的。

dbalreal 函数适用于离散时间系统。

参见：modreal, minreal, gram, ctrb

2. minreal

功能：最小实现与零极点对消。

格式：

```
[am,bm,cm,dm]=minreal(a,b,c,d)
[am,bm,cm,dm]=minreal(a,b,c,d,tol)
[zm,pm]=minreal(z,p)
[zm,pm]=minreal(z,p,tol)
[numm,denm]=minreal(num,den)
[numm,denm]=minreal(num,den,tol)
```

说明：

最小实现是一种模型的实现，它消除了模型中过多的或不必要的状态。对传递函数或

零极点增益模型，这等价于将可彼此对消的零极点对进行对消。

状态空间模型

[am,bm,cm,dm]=minreal(a,b,c,d)可得到状态空间系统(a,b,c,d)的最小实现，屏幕上会显示出所删去的状态数目。

[am,bm,cm,dm]=minreal(a,b,c,d,tol)，可利用误差容限 tol 以确定零极点的对消，当 tol 缺省时，默认为 tol=10 * max(size(a)) * norm(a,1) * eps。

零极点增益模型

[zm,pm]=minreal(z,p)，其中 z 与 p 为零点与极点列矢量，它在 tol=10 * sqrt(eps) * abs(z(i))下消除公共的根。

[zm,pm]=minreal(z,p,tol)可指定 tol。

传递函数模型

[numm,denm]=minreal(num,den)，其中 num 与 den 为传递函数的分子和分母多项式系数，它在 tol=10 * sqrt(eps) * abs(z(i))下消去多项式的公共根。

[numm,denm]=minreal(num,den,tol)可指定 tol。

参见：balreal，modred，ctrbf，obsvf

3. modred，dmodred

功能：模型降阶。

格式：

```
[ar,br,cr,dr]=modred(a,b,c,d,elim)
[ar,br,cr,dr]=dmodred(a,b,c,d,elim)
```

说明：

modred 函数可在保持稳定状态输入/输出关系的前提下，降低状态空间模型的阶次，因此在清除快速高频状态时，modred 函数是非常有用的。利用 ssdelete 可清除低频慢变状态。一般情况下，modred 函数与 balreal 函数配合使用。

[ar,br,cr,dr]=modred(a,b,c,d,elim)，通过消除 elim 中指定的状态来降低模型的阶次。这种操作结果可看作将清除状态设置成无限快状态。

[ar,br,cr,dr]=dmodred(a,b,c,elim)适用于离散时间系统。

参见：balreal，minreal，ssdelete

2.4 模型实现

1. canon

功能：状态空间的正则形式转换。

格式：

```
[ab,bb,cb,db,t]=canon(a,b,c,d,'type')
```

说明：

canon 函数可将连续空间系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

变换成正则形式(type='modal')或伴随正则形式(type='companion')，当参数 type 缺省时，默认为 modal。

转换后的系统与原系统具有相同的输入/输出关系，但状态并不相同。由于伴随正则形式具有病态条件，因此应尽量避免使用这种形式。

[ab,bb,cb,db]=canon(a,b,c,d,'modal')，可将状态空间系统变换成正则形式，其实特征值处于矩阵 A 的对角线上，复特征值处于矩阵 A 对角线的一个 2×2 方阵中。

例 特征值为$(-\lambda_1, \sigma\pm j\omega, \lambda_2)$的正则形式矩阵 A 为

$$A=\begin{bmatrix} \lambda_1 & 0 & 0 & 0 \\ 0 & \sigma & \omega & 0 \\ 0 & -\omega & \sigma & 0 \\ 0 & 0 & 0 & \lambda_2 \end{bmatrix}$$

[ab,bb,cb,db]=canon(a,b,c,d, 'companion')，可将状态空间系统变换成伴随正则形式，其矩阵 A 的最右一列为系统特征多项式的系数。

例 系统的特征多项式为

$$s^n + a_1 s^{n-1} + \cdots + a_{n-1} s + a_n$$

则对应的伴随矩阵 A 为

$$A=\begin{bmatrix} 0 & 0 & 0 & \cdots & -a_n \\ 1 & 0 & 0 & \cdots & \vdots \\ 0 & 1 & 0 & \cdots & -a_3 \\ \vdots & \vdots & \vdots & \ddots & -a_2 \\ 0 & \cdots & \cdots & 1 & -a_1 \end{bmatrix}$$

[ab,bb,cb,db,T]=canon(a,b,c,d, 'type')还可得到变换矢量 T，其中 T 满足 $z=Tx$。

参见：ss2ss，ctrb，ctrbf

2．ctrbf，obsvf

功能：可控性和可观性阶梯形式。

格式：

```
[ab,bb,cb,T,k]=ctrbf(a,b,c)
[ab,bb,cb,T,k]=ctrbf(a,b,c,tol)
[ab,bb,cb,T,k]=obsvf(a,b,c)
[ab,bb,cb,T,k]=obsvf(a,b,c,tol)
```

说明：

当可控性矩阵的秩 $r\leqslant n$(n 为 a 的阶次)时，则存在相似性变换

$$\bar{a} = TaT^T \qquad b = Tb \qquad \bar{c} = cT^T$$

其中 T 为酉的，变换后的系统为阶梯形，如果含有不可控模式，那么它们必然出现在左上角

$$\bar{a} = \begin{bmatrix} a_{uc} & 0 \\ a_{21} & a_c \end{bmatrix}, \quad \bar{b} = \begin{bmatrix} 0 \\ b_c \end{bmatrix}$$

其中(a_c, b_c)为可控部分，(a_{uc}, b_{uc})为不可控制部分，而且有

$$c_c(sI - a_c)^{-1}b_c = c(sI - a)^{-1}b$$

MATLAB 函数[ab,bb,cb,T,k]=ctrbf(a,b,c)可将系统分解为可控/不可控两部分。如上所述，T 为相似性变换，k 是长度为 n 的一个矢量，其元素为各个块的秩。sum(k)可求出 a 中可控部分的秩。

[ab,bb,cb,T,k]=ctrbf(a,b,c,tol)，可指定误差容限，当 tol 缺省时，默认为 tol=10 * n * norm(a,1) * eps。

可观性阶梯形的不可观模式包含在左上角

$$\bar{a} = \begin{bmatrix} a_{uo} & a_{21} \\ 0 & a_o \end{bmatrix}, \quad \bar{c} = \begin{bmatrix} 0 & c_o \end{bmatrix}$$

它可通过

```
[ab,bb,cb,T,k]=obsvf(a,b,c)
[ab,bb,cb,T,k]=obsvf(a,b,c,tol)
```

进行计算。

参见：ctrb，obsv，minreal

3. ss2ss

功能：相似变换。

格式：

```
[at,bt,ct,dt]=ss2ss(a,b,c,d,T)
```

说明：

[at,bt,ct,dt]=ss2ss(a,b,c,d,T)可完成相似变换

$$z = Tx$$

以此得到状态空间系统为

$$\dot{z} = TaT^{-1}z + Tbu$$
$$y = cT^{-1}z + du$$

例　考虑 3 输入/2 输出/4 状态的系统(a,b,c,d)，以第 2 个输出代替第 1 个状态进行相似变换，可输入

```
T=eye(4);
T(1,:)=c(2,:);
[at,bt,ct,dt]=ss2ss(a,b,c,d,T);
```

参见：canon，balreal，balance

2.5 模型特性

1. ctrb，obsv

功能：可控性和可观性矩阵。

格式：

```
co=ctrb(a,b)
ob=obsv(a,c)
```

说明：

ctrb 和 obsv 函数可求出状态空间系统的可控性和可观性矩阵。

对 n×n 矩阵 a，n×m 矩阵 b 和 p×n 矩阵 c，ctrb(a,b)可得到 n×nm 的可控性矩阵

$$co = [b \quad ab \quad a^2b \quad \cdots \quad a^{n-1}b]$$

obsv(a,c)可得到 nm×n 的可观性矩阵

$$ob = \begin{bmatrix} c \\ ca \\ ca^2 \\ \vdots \\ ca^{n-1} \end{bmatrix}$$

当 co 的秩为 n 时，系统可控；当 ob 的秩为 n 时，系统可观。

例 要检查系统(a,b,c,d)的可控性和可观性，可输入

```
co=ctrb(a,b);
% Number of uncontrollable states
unco=length(a)-rank(co)
ob=obsv(a,c);
% Number of unobservable states
unob=length(a)-rank(ob)
```

执行后给出不可控和不可观的状态数。但这种计算可能会包含病态条件，因此最好与 gram，bode，sigma 或 ctrbf 和 obsvf 函数一起来确定系统的可控性和可观性。

参见：ctrbf，obsvf，gram，bode，sigma

2. gram，dgram

功能：求可控和可观的 gram 矩阵。

格式：

```
gc=gram(a,b);
go=gram(a',c');
gc=dgram(a,b);
go=dgram(a',c')
```

说明：

gram 函数可计算出可控与可观的 gram 矩阵，gram 阵可用于研究状态空间系统的可控性与可观性，它们比由 ctrb 与 obsv 形成的可控性与可观性矩阵有更好的特性。gram(a,b)可得到可控 gram 阵

$$gc = \int_0^{\infty} e^{a\tau}bb^T e^{a^T\tau}d\tau$$

gram 阵为对称矩阵，而且当系统可控时，gram 阵为满秩矩阵。

gram(a′,c′)可得到可观 gram 阵

$$go = \int_0^{\infty} e^{a\tau}cc^T e^{a^T\tau}d\tau$$

当系统可观时，gram 阵为满秩矩阵。

dgram 适用于离散时间系统。

例 为评价一个系统的可控性，可输入

```
gc=gram(a,b);
s=svd(gc);
```

参见：balreal，minreal，ctrb，obsv，lyap，rank

3. dcgain，ddcgain

功能：计算系统的稳态(DC)增益。

格式：

```
k=dcgain(a,b,c,d)
k=dcgain(num,den)
k=ddcgain(a,b,c,d)
k=ddcgain(num,den)
```

说明：

dcgain 函数可计算出一个系统的稳态(DC 或低频)增益。

k=dcgain(a,b,c,d)可计算出从所有输入到所有输出的连续状态空间系统的稳态增益

$$k = -ca^{-1}b + d$$

k=dcgain(num,den)可得到多项式传递函数 g(s)=num(s)/den(s)表示的稳态增益

$$k = \left.\frac{num(s)}{den(s)}\right|_{s=0}$$

为计算离散时间系统的 DC 增益，可采用 ddcgain 函数，其 DC 增益矩阵为

$$k = c(I-a)^{-1}b + d$$

$$k = \left.\frac{num(z)}{den(z)}\right|_{z=1}$$

例 计算系统

$$H(s) = \frac{2s^2+5s+1}{s^2+2s+3}$$

的 DC 增益，可输入

```
num=[2  5  1];
```

```
den=[1  2  3];
k=dcgain(num,den)
```

执行后得

```
k=
   0.3333
```

例 计算 MIMO 状态空间系统

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}-0.5572 & -0.7814\\ 0.7814 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}1 & 0.5397\\ 0 & -0.2231\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}$$

$$\begin{bmatrix}y\\ z\end{bmatrix}=\begin{bmatrix}1.9691 & 6.4493\\ 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}0 & 0\\ 0 & 0\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}$$

的 DC 增益,.可输入

```
a=[-0.5572  -0.7814; 0.7814, 0];
b=[ 1  .5397; 0  -.2231];
c=[1.9691  6.4493; 1  0];
d=[0  0; 0  0];
k=dcgain(a,b,c,d)
```

执行后得

```
k=
   8.2535    3.7036
        0    0.2855
```

参见: bode, dbode

4. damp, ddamp

功能:求衰减因子和自然频率。

格式:

```
[Wn,z]=damp(a)
mag=ddamp(a)
[mag,Wn,z]=ddamp(a,Ts)
```

说明:

damp 和 ddamp 函数用于计算自然频率和衰减因子。当不带输出变量时，可在屏幕上显示出特征值表、衰减比率及自然频率。

[Wn,z]=damp(a)可得到连续特征值的自然频率 Wn 和衰减因子 z，变量 a 可以取几种格式:

- 当 a 为方阵，则它为状态空间矩阵 a;
- 当 a 为行矢量，则它为传递函数多项式的系数;
- 当 a 为列矢量，则它为特征根位置值。

mag=ddmap(a)可得到列矢量 mag，它包含从 a 中计算得到的离散特征值的幅值。

[mag,Wn,z]=ddmap(a,Ts)可得到矢量 mag、Wn 和 z，它们分别包含 a 的特征值幅

值、等效 s 平面的自然频率和衰减比率，Ts 为取样时间。离散特征值 λ 的等效 s 平面衰减比率和自然频率为

$$\omega_n = \left|\frac{\log\lambda}{T_s}\right| \qquad \xi = -\cos(\angle\log\lambda)$$

例 对连续系统

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

要计算出特征值、自然频率和衰减比率，可输入

```
num=[2  5  1];
den=[1  2  3];
damp(den);
```

执行后得

Eigenvalue	Damping	Freq. (rad/sec)
−1.0000+1.4142i	0.5774	1.7321
−1.0000−1.4142i	0.5774	1.7321

例 对离散系统

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6z + 0.8}$$

要求其特征值、幅值、等效衰减因子、等效自然频率，可输入

```
num=[2  -3.4  1.5];
den=[1  -1.6  0.8];
damp(den, 0.1);
```

执行后得

Eigenvalue	Magnitude	Equiv. Damping	Equiv. Freq
0.80000+0.4000i	0.8944	0.2340	4.7688
0.80000−0.4000i	0.8944	0.2340	4.7688

参见：无

5. covar, dcovar

功能：白噪声的协方差响应。

格式：

```
[p,q]=covar(a,b,c,d,w)
p=covar(num,den,w)
[p,q]=dcovar(a,b,c,d,w)
p=dcovar(num,den,w)
```

说明：

covar 函数可计算系统输入高斯白噪声(强度为 w)时的稳态输出和状态的协方差响应。

[p,q]=covar(a,b,c,d,w)，可计算出连续状态系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

当输入为白噪声时，输出及状态的协方差响应为

$$p = E[yy^T], \qquad q = E[xx^T]$$

这时系统必须是稳定的，而且矩阵 d 必须为零。

p=covar(num,den,w)，可计算出多项式传递函数 g(s)=num(s)/den(s)表示的 SIMO 系统输出的协方差响应。

计算离散系统的协方差响应可采用 dcovar 函数。

例 SISO 系统

$$H(s) = \frac{5s+1}{s^2+2s+3}$$

要计算输入为高斯白噪声时的协方差响应，这时可输入

```
num=[5  1];
den=[1  2  3];
p=covar(num,den,5);
% check result using simulation
dt=.01   ; t=0 : dt : 10;
w=sqrt(5/dt) * randn(size(t));
y=lsim(num,den,w,t);
plot(t,y), title('Noise Response')
```

执行后得如图 2.15 所示的噪声响应曲线。

参见：lyap，dlyap

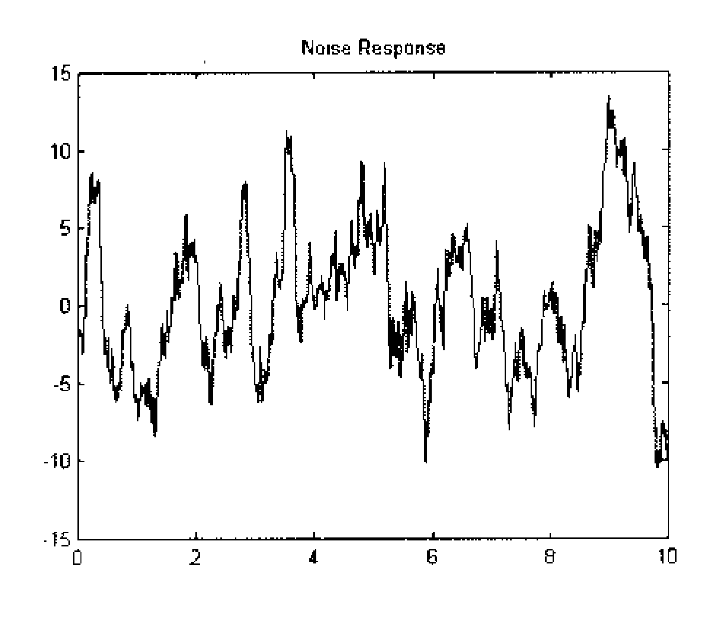


图 2.15　噪声响应曲线

6. esort，dsort

功能：按实部或幅值对特征值进行排序。

格式：

[s,ndx]=esort(p)

[s,ndx]=dsort(p)

说明：

s=esort(p)可根据实部按递减顺序对矢量 p 中的复特征值进行排序。对于连续特征值，先列出不稳定特征值。

s=dsort(p)可根据幅值按递减顺序对矢量 p 中的复特征值进行排序。对于离散特征值，先列出不稳定特征值。

[s,ndx]=esort(p)或[s,ndx]=dsort(p)，还可得到排序中所用到的索引矢量 ndx。

参见： sort

7. tzero

功能： 传递零点。

格式：

z=tzero(a,b,c,d)

说明：

tzero 函数可找出状态空间系统的不变零点，对最小系统而言，不变零点就是传递零点。传递零点是典型 SISO 传递函数零点在多变量系统中的推广，它们相应于非零状态和输入时而输出为零的状态。如下列方程中，复数 λ_2 为不变零点

$$\begin{bmatrix} \dot{x}(t) \\ 0 \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x(t) \\ u_0 e^{\lambda_2 t} \end{bmatrix}$$

tzero(a,b,c,d)可计算出状态空间系统(a,b,c,d)的传递零点，还可以计算状态系统的输入/输出去耦零点。例如，计算输入去耦零点，可输入

zid=tzero(a,b,[ ],[ ])

计算输出去耦零点，可输入

zod=tzero(a, [ ], c, [ ])

参见： ss2zp，pzmap

8. printsys

功能： 显示/打印出线性系统。

格式：

printsys(a,b,c,d)

printsys(a,b,c,d,ulabels,ylabels,xlabels)

printsys(num,den,'s')

printsys(num,den,'z')

说明：

printsys 函数可按特殊格式打印出状态空间和传递函数表示的系统。对于状态空间系统，显示时分别标出输入、输出及其状态，而对传递函数系统，则按多项式之比进行显示。

printsys(a,b,c,d)打印出状态系统(a,b,c,d)并在其上面和左边标记出状态、输入及输出编号。

printsys(a,b,c,d,ulabels,ylabels,xlabels)可用 ulabels、ylabels、xlabels 中指定的符号标记出系统矩阵(a,b,c,d)。例如

```
ylabels=['Phi Theta Psi']
```

表示系统第一个输出标记为 Phi，第二、三个输出分别标记为 Theta 和 Psi。

printsys(num,den,'s')或 printsys(num,den,'z')可打印出两多项式(s 或 z)之比形式的传递函数，参数's'或'z'缺省时，默认为's'。

例 打印出某系统的状态系统，可输入

```
printsys(a,b,c,d)
```

执行后得

a=

	x_1	x_2	x_3
x_1	83.84828	−142.17869	17.26809
x_2	44.32524	−74.14096	8.75416
x_3	−148.14160	249.03473	−32.28152

b=

	u_1
x_1	0.75622
x_2	0.40049
x_3	−1.34138

c=

	x_1	x_2	x_3
y_1	5.46653	25.00495	9.67382
y_2	116.99892	−194.37232	24.20050

d=

	u_1
y_1	−0.11245
y_2	1.02696

如程序改成

```
ulabel=['force']
ylabel=['position velocity']
xlabel=['motor   x   v']
printsys(a,b,c,d,ulabel,ylabel,xlabel)
```

则执行后得

a=

	motor	x	v
motor	83.84828	−142.17869	17.26809
x	44.32524	−74.14096	8.75416

```
        v   -148.14160      249.03473   -32.28152
b=
                    force
   motor          0.75622
       x          0.40049
       v         -1.34138
c=
               motor            x            v
   position     5.46653      25.00495      9.67382
   velocity   116.99892    -194.37232     24.20050
d=
               force
   position    -0.11245
   velocity     1.02696
```

例 要打印某系统的传递函数，可输入

printsys(num,den)

执行后得

```
num/den=
        -0.6734 s^ 3-0.1397 s^ 2+0.7969 s-8.159
               s^ 3+3.881 s^ 2+71.54 s+128
```

参见：无

2.6 方程求解

1. are

功能：代数 Riccati(黎卡堤)方程求解。

格式：

x=are(a,b,c)

说明：

are 函数用于求解代数 Riccati 方程。在控制系统的许多领域如线性二次型调节器和估计器设计等都会涉及到代数 Riccati 方程的求解问题。

x=are(a,b,c)可求出连续时间代数 Riccati 方程的正定解(如果存在的话)

$$a^{T}x + xa + xbx + c = 0$$

解的有效性可用函数 ric 验证。

参见：lqe，lqr，schur，schord，ric

2. lyap，lyap2，dlyap

功能：Lyapunov(李亚普诺夫)方程求解。

格式：

```
x=lyap(a,b,c)
x=lyap(a,c)
x=dlyap(a,c)
```

说明：

lyap 函数可求解一般形式或特殊形式的 Lyapunov 方程。在研究控制系统的稳定性、RMS 性能时都会涉及到 Lyapunov 方程。

lyap(a,b,c)可求解连续系统的一般形式的 Lyapunov 方程

$$ax + xb = -c$$

其中 a,b,c 为适当维数的已知矩阵。

lyap(a,c)可求解特殊形式的 Lyapunov 方程

$$ax + xa^T = -c$$

dlyap(a,b,c)或 dlyap(a,c)用于求解离散系统的 Lyapunov 方程。

lyap2 函数类似于 lyap，只是采用了特征值分解技术求解 Lyapunov 方程，因此运算速度大大加快。

例 输入

```
a=[2,3,2; 1,2,-1; 3,-4,5];
c=[1,0,0; 0,1,0; 0,0,1];
x=lyap(a,c)
```

执行后得

```
x=
      0.4263    -0.1842    -0.4000
     -0.1842    -1.1031    -0.1097
     -0.4000     0.1097     0.2278
```

这可通过 a * x+x * a' 加以验证。

参见：covar，dcovar，gram，dgram

2.7 时域响应

1. step

功能：求连续系统的单位阶跃响应。

格式：

```
[y,x,t]=step(a,b,c,d)
[y,x,t]=step(a,b,c,d,iu)
[y,x,t]=step(a,b,c,d,iu,t)
[y,x,t]=step(num,den)
[y,x,t]=step(num,den,t)
```

说明：

step 函数可计算出线性系统的单位阶跃响应，当不带输出变量引用时，step 函数可在当前图形窗口中绘出系统的阶跃响应曲线。

step(a,b,c,d)可得到一组阶跃响应曲线，每条曲线对应于连续 LTI 系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

的输入/输出组合，其时间矢量由函数自动设定。

step(a,b,c,d,iu)可绘制出从第 iu 个输入到所有输出的单位阶跃响应曲线。

step(a,b,c,d,iu,t)或 step(num,den,t)可利用用户指定的时间矢量 t 来绘制单位阶跃响应。

当带有输出变量引用函数时，可得到系统阶跃响应的输出数据，而不直接绘制出曲线。

例 有一二阶系统

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} \begin{bmatrix} -0.5572 & -0.7814 \\ 0.7814 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

$$y = \begin{bmatrix} 1.9691 & 6.4493 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]\, u$$

要求出系统的阶跃响应，可输入

```
a=[-0.5572  -0.7814; 0.7814  0];
b=[1; 0];
c=[1.9691  6.4493];
d=[ 0 ];
step(a,b,c,d);
title( 'Step Response')
```

执行后得到如图 2.16 所示的单位阶跃响应曲线。

参见：lsim，initial，impulse，dstep

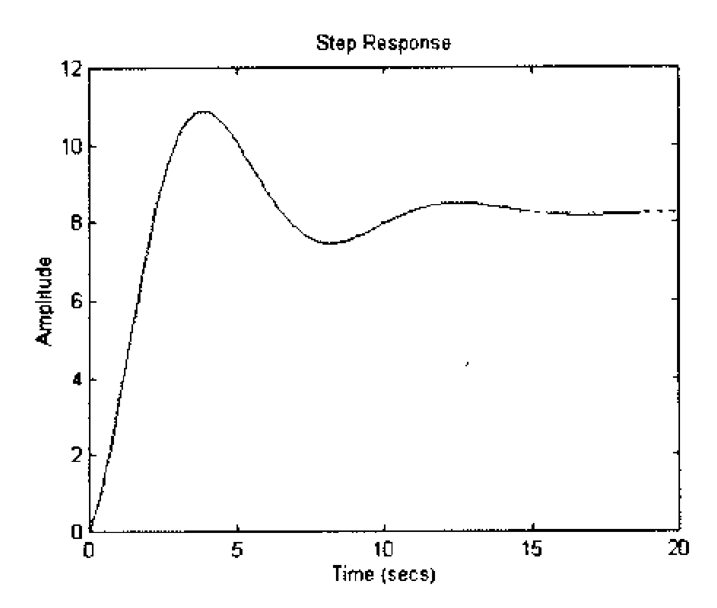


图 2.16 连续系统的单位阶跃响应

2. dstep

功能：求离散系统的单位阶跃响应。

格式：

```
[y,x]=dstep(a,b,c,d)
[y,x]=dstep(a,b,c,d,iu)
[y,x]=dstep(a,b,c,d,iu,n)
[y,x]=dstep(num,den)
[y,x]=dstep(num,den,n)
```

说明：

dstep 函数可计算出离散时间线性系统的单位阶跃响应，当不带输出变量引用时，dstep 可在当前图形窗口中绘出系统的阶跃响应曲线。

dstep(a,b,c,d)可得到一组阶跃响应曲线，每条曲线对应于离散 LTI 系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

的输入/输出组合对，其取样点数自动选取。

dstep(a,b,c,d,iu)可绘制出从第 iu 个输入到所有输出的阶跃响应曲线。

dstep(num,den)可绘制出以多项式传递函数 g(z)=num(z)/den(z)表示的系统阶跃响应曲线。

dstep(a,b,c,d,iu,n)或 dstep(num,den,n)可利用用户指定的取样点数来绘制系统的单位阶跃响应曲线。

当带输出变量引用函数时，可得到系统阶跃响应的输出数据，而不直接绘制出曲线。

例 某二阶系统

$$h(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6z + 0.8}$$

要求其阶跃响应，可输入

```
num=[2  -3.4  1.5];
den=[1  -1.6  0.8];
dstep(num,den)
title('Discrete Step Response')
```

执行后得到如图 2.17 所示的阶跃响应曲线。

参见：step，dlsim，dimpulse，dinitial

3. impulse

功能：求连续系统的单位冲激响应。

格式：

```
[y,x,t]=impulse(a,b,c,d)
[y,x,t]=impulse(a,b,c,d,iu)
```

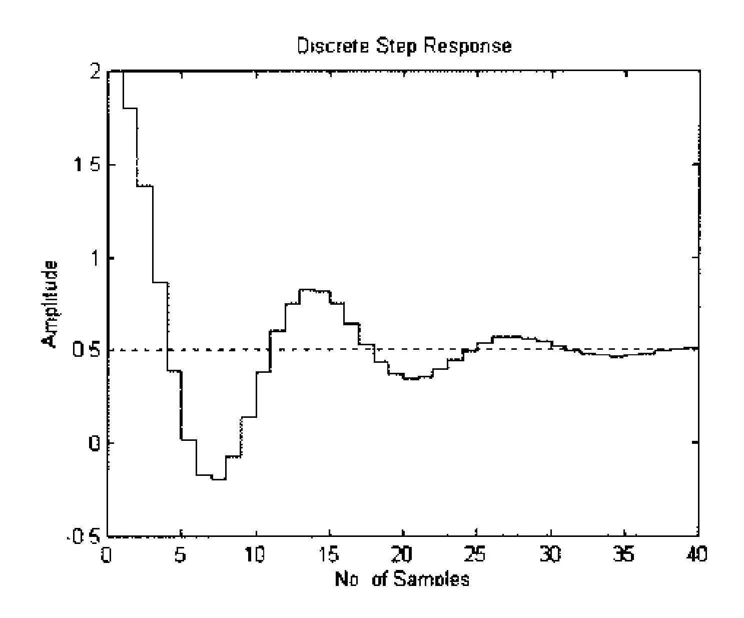


图 2.17　离散系统的阶跃响应曲线

```
[y,x,t]=impulse(a,b,c,d,iu,t)
[y,x,t]=impulse(num,den)
[y,x,t]=impulse(num,den,t)
```

说明：

impulse 函数用于计算线性系统的单位冲激响应，当不带输出变量时，impulse 可在当前图形窗口中直接绘出系统的单位冲激响应。

impulse(a,b,c,d)可得到一组冲激响应曲线，每条曲线对应于连续 LTI 系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

的输入/输出组合对，而时间参量自动选取。

impulse(a,b,c,d,iu)可绘制出从第 iu 个输入到所有输出的冲激响应曲线。

impulse(num,den)可绘制出以多项式传递函数 g(s)=num(s)/den(s)表示的系统冲激响应曲线。

impulse(a,b,c,d,iu,t)或 impulse(num,den,t)可利用用户指定的时间矢量 t 来绘制冲激响应曲线。

当带输出变量引用函数时，可得到系统单位冲激响应的输出数据，而不直接绘制出曲线。

例　有一二阶系统

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}\begin{bmatrix}-0.5572 & -0.7814\\ 0.7814 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}1\\ 0\end{bmatrix}u$$

$$y = \begin{bmatrix}1.9691 & 6.4493\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}0\end{bmatrix}u$$

要求出系统的冲激响应，可输入

```
a=[-0.5572   -0.7814; 0.7814   0];
b=[1; 0];
```

```
c=[1.9691   6.4493];
d=[ 0 ];
impulse(a,b,c,d)
title( 'Impulse Response')
```

执行后得到如图 2.18 所示的冲激响应曲线。

参见：dimpulse，lsim，step，initial

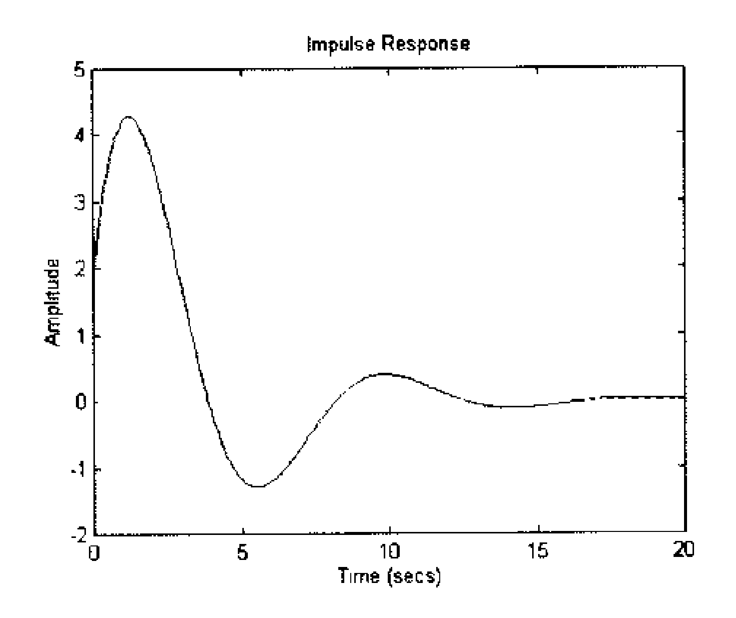


图 2.18　连续系统的冲激响应曲线

4. dimpulse

功能：求离散系统的单位冲激响应。

格式：

```
[y,x]=dimpulse(a,b,c,d)
[y,x]=dimpulse(a,b,c,d,iu)
[y,x]=dimpulse(a,b,c,d,iu,n)
[y,x]=dimpulse(num,den)
[y,x]=dimpulse(num,den,n)
```

说明：

dimpulse 函数用于计算离散时间线性系统的单位冲激响应，当不带输出变量引用函数时，dimpulse 可在当前图形窗口中绘出系统的冲激响应曲线。

dimpulse(a,b,c,d)可得到一组冲激响应曲线，每条曲线对应于离散 LTI 系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

的输入/输出组合对，而取样点数自动选取。

dimpulse(a,b,c,d,iu)可绘制出从第 iu 个输入到所有输出的冲激响应曲线。

dimpulse(num,den)可绘制出以多项式传递函数 g(z)=num(z)/den(z)表示的系统冲

激响应曲线，其中 num 和 den 为按 z 的递减幂次排列的多项式系数。

dimpulse(a,b,c,d,iu,n)或 impulse(num,den,n)可利用用户指定的取样点 n 来绘制系统的冲激响应曲线。

当带输出变量引用函数时，可得到系统单位冲激响应的输出数据，而不直接绘制出曲线。

例 有一系统

$$H(z)=\frac{2s^2-3.4z+1.5}{z^2-1.6z+0.8}$$

要求出系统的冲激响应，可输入

```
num=[2  -3.4  1.5];
den=[1  -1.6  0.8];
dimpulse(num,den)
title('Discrete Impulse Response')
```

执行后可得如图 2.19 所示的冲激响应曲线。

参见：impulse，dlsim，dstep，dinitial

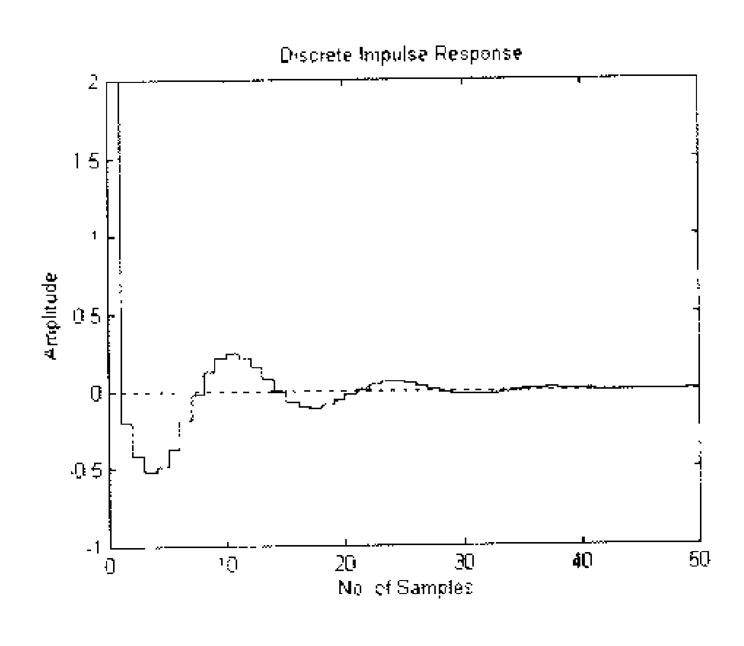


图 2.19 离散系统的冲激响应曲线

5. initial

功能：求连续系统的零输入响应。

格式：

```
[y,x,t]=initial(a,b,c,d,x0)
[y,x,t]=initial(a,b,c,d,x0,t)
```

说明：

initial 函数可计算出连续时间线性系统由于初始状态所引起的响应(故而称零输入响应)。当不带输出变量引用函数时，initial 函数在当前图形窗口中直接绘制出系统的零输入响应。

initial(a,b,c,d,x0)可绘出连续时间 LTI 系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

每一个输出的零输入响应曲线，x_0 为初始状态，时间矢量由函数自动选取。

initial(a,b,c,d,x0,t)可利用指定的时间矢量 t 来绘制零输入响应曲线。

当带输出变量引用函数时，可得到系统零输入响应的输出数据，而不直接绘制出曲线。

例 有一二阶系统

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix}=\begin{bmatrix}-0.5572 & -0.7814\\ 0.7814 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}1\\ 0\end{bmatrix}u$$

$$y=\begin{bmatrix}1.9691 & 6.4493\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}+\begin{bmatrix}0\end{bmatrix}u$$

当初始状态 x0=[1；0]时，求零输入响应，这时可输入

```
a=[-0.5572  -0.7814; 0.7814  0];
b=[1; 0];
c=[1.9691  6.4493];
d=[ 0 ];
xo=[1; 0];
t=0 : 0.1 : 20;
initial(a,b,c,d,x0,t)
title( 'Initial Condition Response')
```

执行后得到如图 2.20 所示的零输入响应曲线。

参见：dinitial，lsim，step，impulse

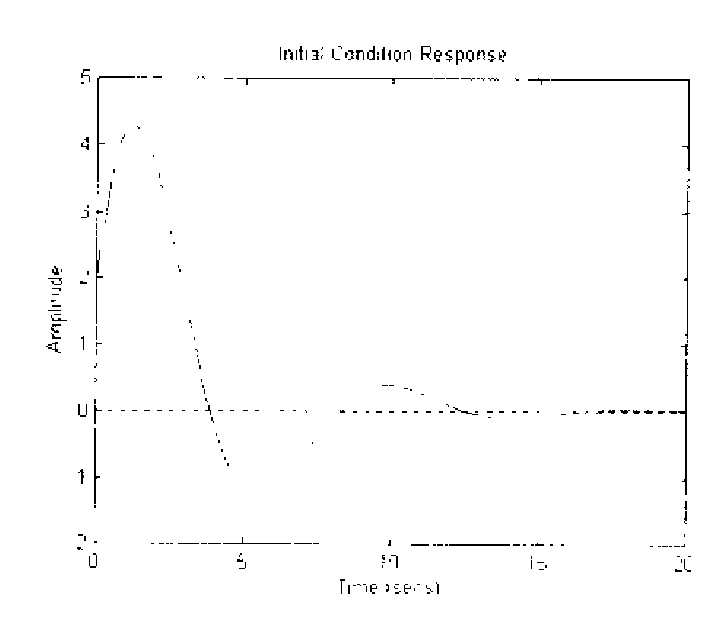


图 2.20 连续系统的零输入响应

6. dinitial

功能：求离散系统的零输入响应。

格式：

```
[y,x]=dinitial(a,b,c,d,x0)
[y,x]=dinitial(a,b,c,d,x0,n)
```

说明：

dinitial 函数可计算出离散时间线性系统由于初始状态所引起的响应(故而称零输入响应)。当不带输出变量引用函数时，dinitial 可在当前图形窗口中直接绘制出系统的零输入响应。

dinitial 可绘出离散时间 LTI 系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

每一个输出的零输入响应，x0 为初始状态，取样点数由函数自动选取。

dinitial(a,b,c,d,x0,n)可利用指定的取样点数 n 来绘制零输入响应。

当带输出变量引用函数时，可得到系统零输入响应的输出数据，而不直接绘制出曲线。

例 二阶系统

$$\begin{bmatrix} x_1[n+1] \\ x_2[n+1] \end{bmatrix} = \begin{bmatrix} -.7497 & -.2027 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix} + \begin{bmatrix} -4.1841 \\ -6.5049 \end{bmatrix} u$$

$$y = [3.9321 \quad 0] \begin{bmatrix} x_1[n] \\ x_2[n] \end{bmatrix}$$

当初始状态为 x0=[1；0]时，求系统的零输入响应，这时可输入

```
a=[-.7497, -.2027; 1,0];
b=[-4.1841; -6.5049];
c=[3.9321, 0];
d=[0];
x0=[1; 0];
dinitial(a,b,c,d,x0);
title('Discrete Initial Condition Response')
```

执行后得到如图 2.21 所示的零输入响应曲线。

参见：initial，dlsim，dstep，dimpulse

7. lsim

功能：对任意输入的连续系统进行仿真。

格式：

```
[y,x]=lsim(a,b,c,d,u,t)
[y,x]=lsim(a,b,c,d,u,t,x0)
[y,x]=lsim(num,den,u,t)
```

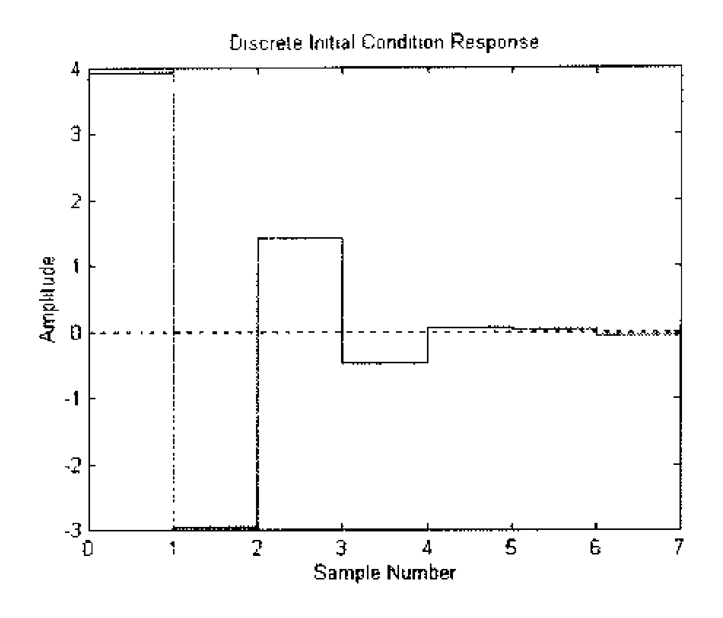


图 2.21 离散系统的零输入响应

说明：

lsim 函数可对任意输入的连续时间线性系统进行仿真，在不带输出变量引用函数时，lsim 可在当前图形窗口中绘制出系统的输出响应曲线。

给定 LTI 系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

lsim(a,b,c,d,u,t)函数可针对输入 u 绘出系统的输出曲线，其中 u 中给出每个输入的时序列，因此一般情况下 u 应为矩阵；t 用于指定仿真的时间轴，它应为等间隔。

lsim(a,b,c,d,u,t,x0)还给出了系统的初始状态 x0。

lsim(num,den, u,t)以传递函数 g(s)＝num(s)/den(s)形式指定仿真系统。

当带输出变量引用函数时，可得到系统输出响应曲线的数据，而不直接绘制出曲线。

例 二阶系统

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

现要求出周期为 4 秒的方波输出响应，可输入

```
num=[2  5  1];
den=[1  2  3];
t=(0 : .1 : 10);
period=4;
u=(rem(t,period)>=period. /2);
lsim(num,den,u,t);
title( 'Square ware response')
```

执行后得到如图 2.22 所示的输出响应曲线。

参见：impulse, step, dimpulse, dstep, dlsim, filter, c2d. ltitr、initial

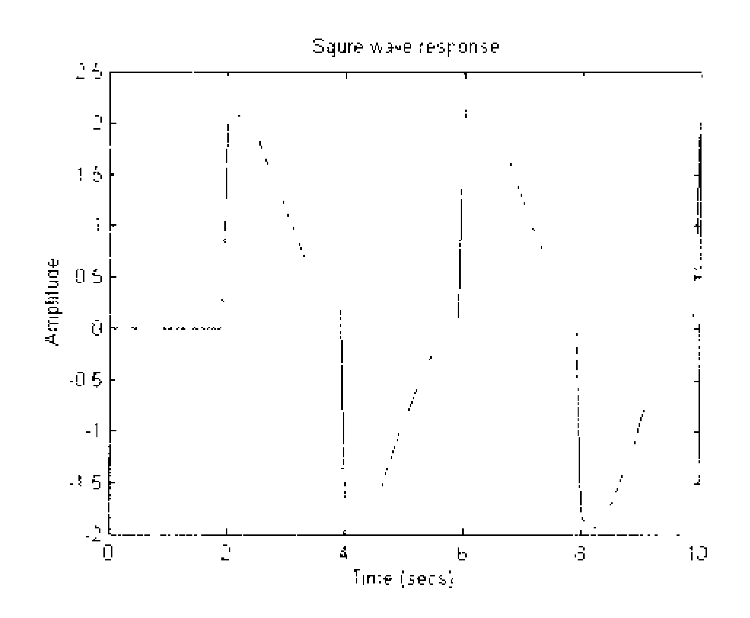


图 2.22　连续系统的输出响应

8. dlsim

功能：对任意输入的离散系统进行仿真。

格式：

```
[y,x]=dlsim(a,b,c,d,u)
[y,x]=dlsim(a,b,c,d,u,x0)
[y,x]=dlsim(num,den,u)
```

说明：

dlsim 函数可对任意输入的离散时间线性系统进行仿真。在不带输出变量引用函数时，dlsim 可在当前图形窗口中直接绘制出系统的输出曲线。

给定 LTI 系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

dlsim(a,b,c,d,u)函数可针对输入 u 绘出系统的输出曲线，其中 u 给出每个输入的取样值。

dlsim(a,b,c,d,u,x0)还给出了系统的初始状态 x0。

dlsim(num,den,u)以传递函数 g(z)=num(z)/den(z)形式指定仿真系统。

当带输出变量引用函数时，可得到系统输出响应曲线的数据，而不直接绘制出曲线。

例　二阶系统

$$H(z) = \frac{2z^2 - 3.4z + 1.5}{z^2 - 1.6z + 0.8}$$

现要求出系统对 100 点随机噪声的响应曲线，可输入

```
num=[2  -3.4  1.5];
den=[1    1.6  0.8];
u=rand(100,1);
dlsim(num,den,u);
```

title('Noise Response')

执行后得到如图 2.23 所示的输出响应曲线。

参见：lsim，dimpulse，dstep，impulse，step，filter

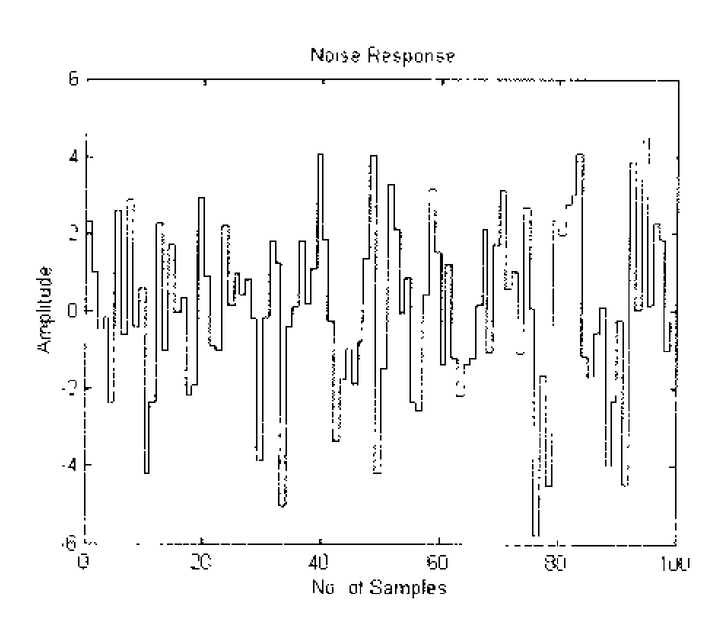


图 2.23　离散系统的输出响应

9. ltitr

功能：求线性时不变时间响应。

格式：

ltitr(a,b,u)

ltitr(a,b,u,x0)

说明：

ltitr 函数涉及线性时不变状态空间系统时间响应的计算核，通常不直接采用它，它可对离散时间状态空间系统进行仿真。

x=ltitr(a,b,u)可得到离散时间系统

$$x[n+1]=ax[n]+bu[n]$$

在输入 u 作用下的状态 x。

x=ltitr(a,b,u,x0)可指定初始状态 x0。

ltitr 函数由 impulse，step，lsim，dimpulse，dstep 及 dlsim 函数引用。

参见：impulse，step，lsim，dimpulse，dstep，dlsim，filter

2.8　频域响应

1. bode

功能：求连续系统的 Bode(波特)频率响应。

格式：

```
[mag,phase,w]=bode(a,b,c,d)
[mag,phase,w]=bode(a,b,c,d,iu)
[mag,phase,w]=bode(a,b,c,d,iu,w)
[mag,phase,w]=bode(num,den)
[mag,phase,w]=bode(num,den,w)
```

说明：

bode 函数可计算出连续时间 LTI 系统的幅频和相频响应曲线(即 Bode 图)。Bode 图可用于分析系统的增益裕度、相位裕度、直接增益、带宽、扰动抑制及其稳定性等特性。当缺省输出变量时，bode 函数可在当前图形窗口中直接绘制出 LTI 系统的 Bode 图。

bode(a,b,c,d)可绘制出系统的一组 Bode 图，它们是针对连续状态空间系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

的每个输入的 Bode 图。其中频率范围由函数自动选取，而且在响应快速变化的位置会自动采用更多取样点。

bode(a,b,c,d,iu)可得到从系统第 iu 个输入到所有输出的 Bode 图。

bode(num,den)可绘制出以连续时间多项式传递函数 g(s)=num(s)/den(s)表示的系统 Bode 图。

bode(a,b,c,d,iu,w)或 bode(num,den,w)，可利用指定的频率矢量绘制出系统的 Bode 图。

当带输出变量引用函数时，可得到系统 Bode 图相应的幅度、相位及频率点矢量，其相互关系为

$$g(s) = c(sI - a)^{-1}b + d$$
$$mag(\omega) = |g(j\omega)|$$
$$phase(\omega) = \angle g(j\omega)$$

相位以度为单位，幅度可转换成分贝为单位

```
magdb=20*log10(mag)
```

对对角化系统可采用 fbode 函数绘制 Bode 图，它采用了基于系统对角矩阵 a 的快速算法，以此提高运算速度。

例 有一二阶系统，其自然频率 $\omega_n=1$，阻尼因子 $\xi=0.2$，要绘制出系统的幅频和相频曲线，可输入

```
[a,b,c,d]=ord2(1,.2);
bode(a,b,c,d);
title('Bode Plot')
```

执行后得到如图 2.24 所示的 Bode 图。

参见： logspace, dbode, margin, nichols, nyquist

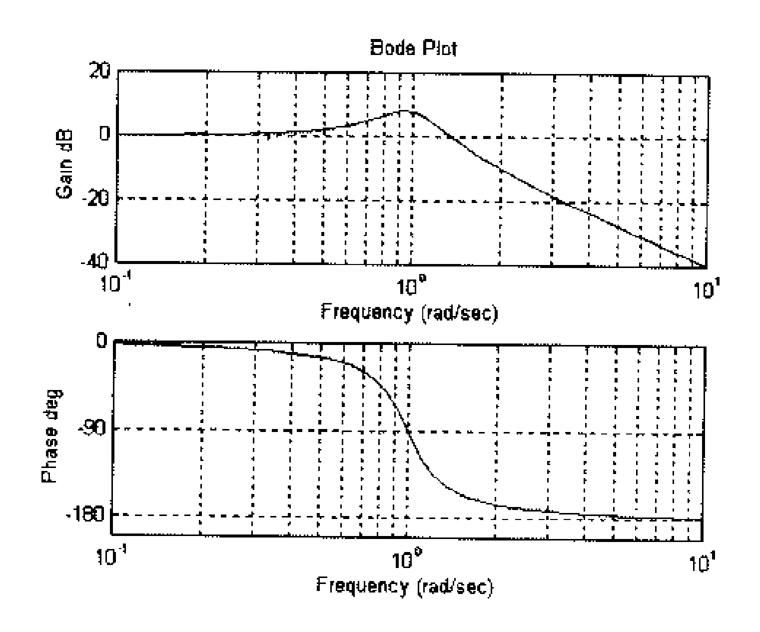


图 2.24　连续系统的 Bode 图

2. dbode

功能：求离散系统的 Bode 频率响应。

格式：

```
[mag,phase,w]=dbode(a,b,c,d,Ts)
[mag,phase,w]=dbode(a,b,c,d,Ts,iu)
[mag,phase,w]=dbode(a,b,c,d,Ts,iu,w)
[mag,phase,w]=dbode(num,den,Ts)
[mag,phase,w]=dbode(num,den,Ts,w)
```

说明：

dbode 函数用于计算离散时间 LTI 系统的幅频和相频响应(即 Bode 图)，当不带输出变量引用函数时，dbode 函数可在当前图形窗口中直接绘制出系统的 Bode 图。

dbode(a,b,c,d,Ts)可得到一组 Bode 曲线，每条曲线对应于离散状态空间系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

的每个输入，其频率范围由函数自动选取。Ts 为取样频率，频率点在 0 到 π/Ts 间选取。在响应快速变化的位置自动选取更多的取样点。

dbode(a,b,c,d,Ts,iu)可得到从第 iu 个输入到所有输出的系统 Bode 图。

dbode(num,den,Ts)可得到以离散时间多项式传递函数 g(z)=num(z)/den(z)表示的系统 Bode 图。

dbode(a,b,c,d,Ts,iu,w)或 abode(num,den,Ts,w)可利用指定的频率范围 w 来绘制系统的 Bode 图。

当带输出变量引用函数时，可得到系统 Bode 图的数据，而不直接绘制出 Bode 图，幅值和相位可根据以下公式计算

$$g(z)=c(zI-a)^{-1}b+d$$
$$mag(\omega)=|g(e^{j\omega t})|$$
$$phase(\omega)=\angle g(e^{j\omega t})$$

其中 t 为内部取样时间，相位以度为单位，幅值可以以分贝为单位表示

magdb=20 * log10(mag)

例 有一二阶系统

$$H(z)=\frac{2z^2-3.4z+1.5}{z^2-1.6z+0.8}$$

要绘制出 Bode 图(设 Ts=0.1)，则可输入

```
num=[2  -3.4  1.5];
den=[1  -1.6  0.8];
dbode(num,den,0.1);
subplot(2,1,1);
title('Discrete Bode Plot')
```

执行后得到如图 2.25 所示的 Bode 图。

参见：loyspace，bode，dnyquist，dnichols，margin

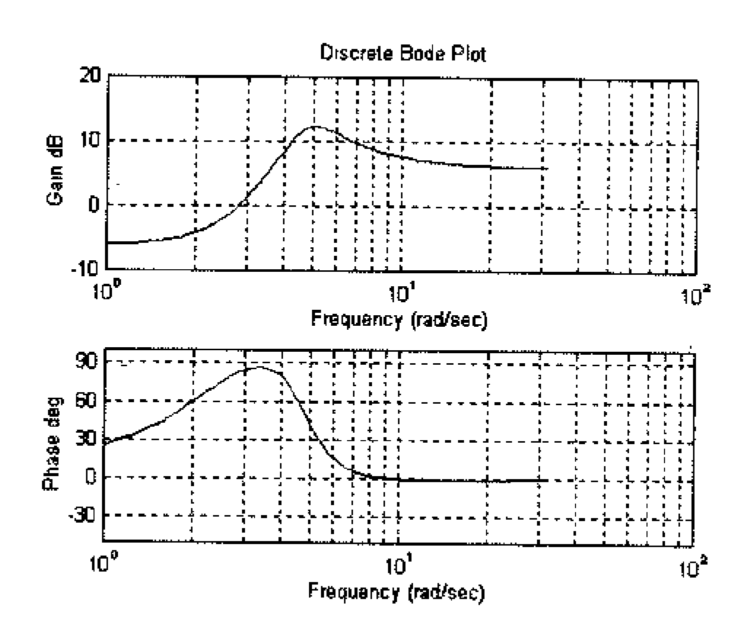


图 2.25 离散系统的 Bode 图

3. nyquist

功能：求连续系统的 Nyquist(奈奎斯特)频率曲线。

格式：

```
[re,im,w]=nyquist(a,b,c,d)
[re,im,w]=nyquist(a,b,c,d,iu)
[re,im,w]=nyquist(a,b,c,d,iu,w)
[re,im,w]=nyquist(num,den)
```

[re,im,w]=nyquist(num,den,w)

说明：

nyquist 函数可计算连续时间 LTI 系统的 Nyquist 频率曲线，Nyquist 曲线可用来分析包括增益裕度、相位裕度及稳定性在内的系统特性。当不带输出变量引用函数时，nyquist 函数会在当前图形窗口中直接绘制出 Nyquist 曲线。

nyquist 函数可以确定单位负反馈系统的稳定性。给定开环系统传递函数 g(s)的 Nyquist 曲线，如果 Nyquist 曲线按逆时针方向包围−1+j0 点 p 次(p 为不稳定开环极点数)，则闭环系统

$$g_{cl}(s) = \frac{g(s)}{1 + g(s)}$$

是稳定的。

nyquist(a,b,c,d)可得到一组 Nyquist 曲线，每条曲线相应于连续状态空间系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$

的输入/输出组合对，其频率范围由函数自动选取，而且在响应快速变化的位置自动选取更多的取样点。

nyquist(a,b,c,d,iu)可得到从第 iu 个输入到系统所有输出的 Nyquist 曲线。

nyquist(num,den)可得到连续多项式传递函数 g(s)=num(s)/den(s)表示的系统 Nyquist 曲线。

nyquist(a,b,c,d,iu,w)或 nyquist(num,den,w)可利用指定的频率向量 w 来绘制系统的 Nyquist 曲线。

当带输出变量引用函数时，可得到系统 Nyquist 曲线的数据，而不直接绘制出系统的 Nyquist 曲线。

例　有二阶系统

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

现要得到系统的 Nyquist 曲线，可输入

```
num=[2  5  1];
den=[1  2  3];
nyquist(num,den);
title( 'Nyquist Plot')
```

执行后得到如图 2.26 所示的结果曲线。由于曲线没有包围−1+j0 点且 p=0，所以由H(s)单位负反馈构成的闭环系统稳定。

参见：dnyquist，nichols，bode，logspace

4. dnyquist

功能：求离散系统的 Nyquist(奈奎斯特)频率曲线。

格式：

[re,im,w]=dnyquist(a,b,c,d,Ts)

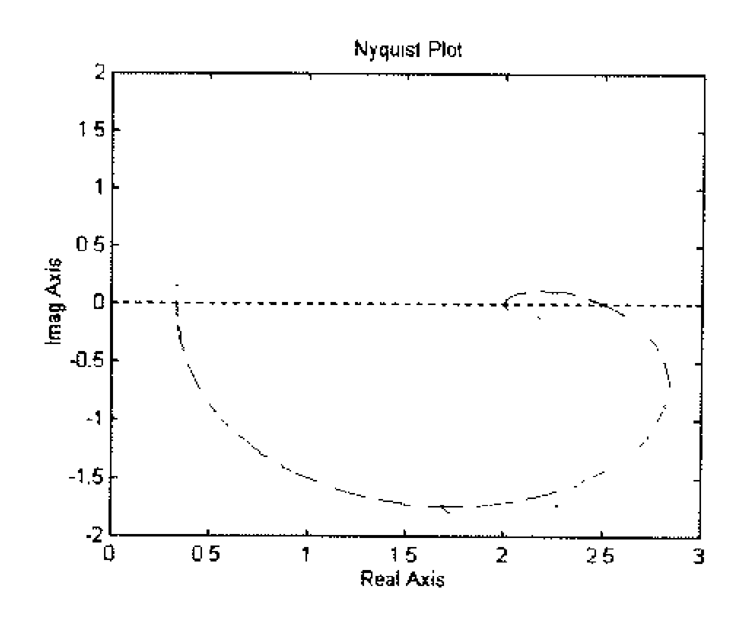


图 2.26 连续系统的 Nyquist 曲线

```
[re,im,w]=dnyquist(a,b,c,d,Ts,iu)
[re,im,w]=dnyquist(a,b,c,d,Ts,iu,w)
[re,im,w]=dnyquist(num,den,Ts)
[re,im,w]=dnyquist(num,den,Ts,w)
```

说明：

dnyquist 函数可计算出离散时间 LTI 系统的 Nyquist 频率响应曲线，当不带输出变量引用函数时，dnyquist 可在当前图形窗口中直接绘制出系统的 Nyquist 曲线。

dnyquist 函数可以确定单位负反馈系统的稳定性。给定开环系统传递函数 g(z)的 Nyquist 曲线，如果 Nyquist 曲线按逆时针方向围绕－1＋j0 点 p 次(p 为不稳定开环极点数)，则闭环传递函数

$$g_{cl}(z) = \frac{g(z)}{1 + g(z)}$$

是稳定的。

dnyquist(a,b,c,d,Ts)可得到一组 Nyquist 曲线，每条曲线对应于离散状态空间系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

的输入/输出组合对，其中频率范围自动选取，频率点在 0 到 π/Ts 弧度之间选取，在响应快速变化的位置会自动选取更多的取样点，Ts 是内部取样时间。

dnyquist(a,b,c,d,Ts,iu)可得到从第 iu 输入到系统所有输出的 Nyquist 曲线。

dnyquist(num,den,Ts)可得到的离散多项式传递函数 g(z)＝num(z)/den(z)表示的系统 Nyquist 曲线。

dnyquist(a,b,c,d,Ts,iu,w)或 dnyquist(num,den,Ts,w)可利用指定的频率矢量 w 来绘制系统的 Nyquist 曲线。

当带输出变量引用函数时，可得到系统 Nyquist 曲线的数据，而不直接绘制出 Nyquist 曲线。

例 有二阶系统

$$H(z)=\frac{2z^2-3.4z+1.5}{z^2-1.6z+0.8}$$

现要计算其 Nyquist 曲线，可输入

```
num=[2  -3.4  1.5];
den=[1  -1.6  0.8];
dnyquist(num,den,0.1);
title('Discrete Nyquist Plot')
```

执行后得到如图 2.27 所示的结果曲线。由于曲线没有包围−1+j0 点且 p=0，因此由H(z) 单位负反馈构成的闭环系统是稳定的。

参见：nyquist，dnichols，dbode，logspace

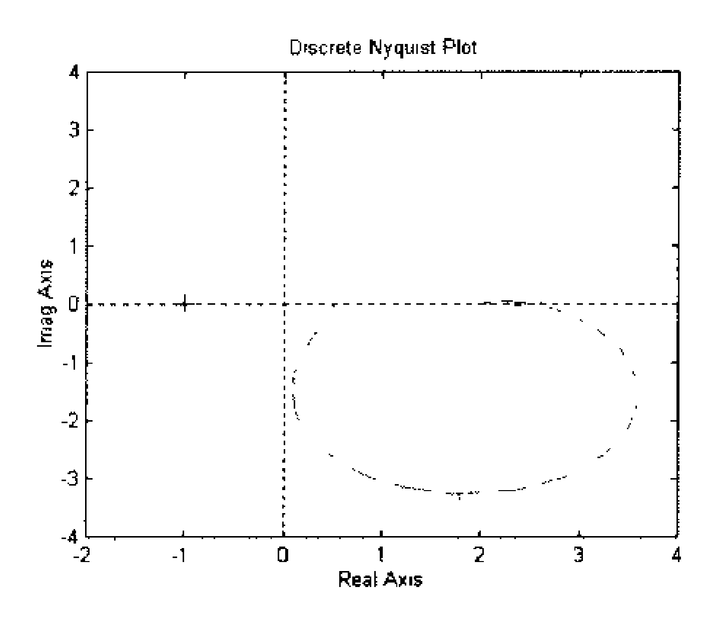


图 2.27 · 离散系统的 Nyquist 曲线

5. nichols

功能：求连续系统的 Nichols(尼柯尔斯)频率响应曲线。

格式：

```
[mag,phase,w]=nichols(a,b,c,d)
[mag,phase,w]=nichols(a,b,c,d,iu)
[mag,phase,w]=nichols(a,b,c,d,iu,w)
[mag,phase,w]=nichols(num,den)
[mag,phase,w]=nichols(num,den,w)
```

说明：

nichols 函数可计算连续时间 LTI 系统的 Nichols 频率响应曲线，Nichols 曲线可用于分析开环和闭环系统的特性。当不带输出变量引用函数时，nichols 可在当前图形窗口中直接绘制出系统的 Nichols 曲线。

nichols(a,b,c,d)可得到一组 Nichols 曲线，每条曲线对应于连续状态空间系统

$$\dot{x} = ax + bu$$
$$y = cx + du$$
的一个输入/输出组合对，其频率范围自动选取，在响应快速变化的位置会自动选取更多的取样点。

nichols(a,b,c,d,iu)可得到从第 iu 个输入到系统所有输出间的 Nichols 曲线。

nichols(num,den)可得到以连续多项式传递函数 g(s)=num(s)/den(s)表示的系统 Nichols 曲线。

nichols(a,b,c,d,iu,w)或 nichols(num,den,w)可利用指定的频率矢量 w 来绘制系统的 Nichols 曲线。

当带输出变量引用函数时，可得到系统 Nichols 曲线的数据，而不直接绘制出 Nichols 曲线，其幅值和相位可由下式得到

$$g(s) = c(sI - a)^{-1}b + d$$
$$mag(\omega) = |g(j\omega)|$$
$$phase(\omega) = \angle g(j\omega)$$
利用 ngrid 函数还可以在 Nichols 曲线上绘制出网格。

例 有四阶系统

$$H(s) = \frac{-4s^4 + 48s^3 - 18s^2 + 250s + 600}{s^4 + 30s^3 + 282s^2 + 525s + 60}$$
现要绘制其 Nichols 曲线，可输入

```
num=[-4  48  -18  250  600];
den=[1  30  282  525  60];
ngrid('new');
nichols(num,den);
title('Nichols Plot')
```
执行后得到如图 2.28 所示的结果曲线。

参见：ngrid，bode，nyquist，dnichols，freqs，logspace，margin

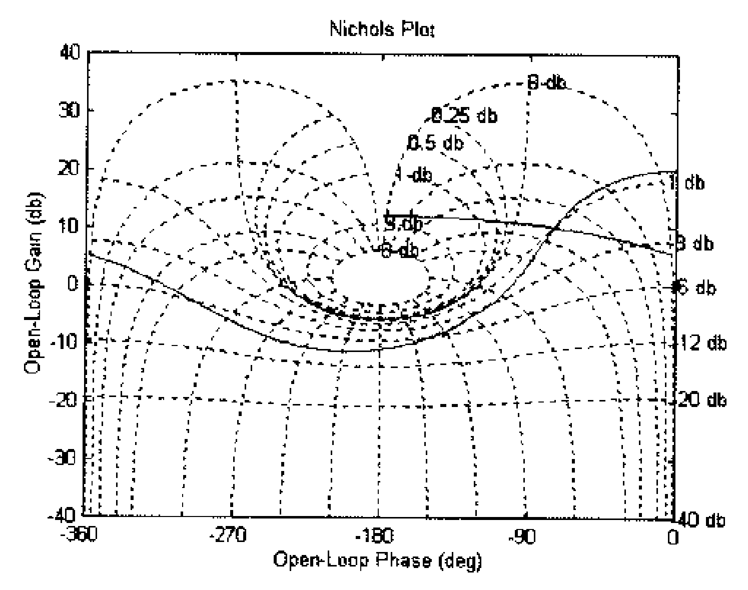


图 2.28 连续系统的 Nichols 曲线

6. dnichols

功能：求离散系统的 Nichols(尼柯尔斯)频率响应曲线。

格式：

```
[mag,phase,w]=dnichols(a,b,c,d,Ts)
[mag,phase,w]=dnichols(a,b,c,d,Ts,iu)
[mag,phase,w]=dnichols(a,b,c,d,Ts,iu,w)
[mag,phase,w]=dnichols(num,den,Ts)
[mag,phase,w]=dnichols(num,den,Ts,w)
```

说明：

dnichols 函数可计算出离散时间 LTI 系统的 Nichols 频率响应曲线，Nichols 曲线可用于分析开环和闭环系统的特性。当不带输出变量引用函数时，dnichols 可在当前图形窗口中直接绘制出系统的 Nichols 曲线。

dnichols(a,b,c,d,Ts)可得到一组 Nichols 曲线，每条曲线对应于离散状态空间系统

$$x[n+1] = ax[n] + bu[n]$$
$$y[n] = cx[n] + du[n]$$

的输入/输出组合对，其频率在 0 到 π/Ts 弧度间选取，频率范围自动选取，在响应快速变化的位置会自动选取更多的取样点，Ts 是内部取样时间。

dnichols(a,b,c,d,Ts,iu)可得到从第 iu 个输入到系统所有输出间的 Nichols 曲线。

dnichols(num,den,Ts)可得到离散多项式传递函数 g(z)=num(z)/den(z)表示的系统 Nichols 曲线。

dnichols(a,b,c,d,Ts,iu,w)或 dnichols(num,den,Ts,w)可利用指定的频率矢量 w 来绘制系统的 Nichols 曲线。

当带输出变量引用函数时，可得到系统 Nichols 曲线的数据，而不直接绘出 Nichols 曲线，其中幅值和相位可由下式得到

$$g(z) = c(zI - a)^{-1}b + d$$
$$mag(\omega) = |g(e^{j\omega t})|$$
$$phase(\omega) = \angle g(e^{j\omega t})$$

其中 t 是内部取样时间。

利用函数 ngrid 可在 Nichols 曲线上绘制出网格。

例 有四阶系统

$$H(z) = \frac{1.5}{z^4 + 1.1z^3 + 1.36z^2 + .88z + .31}$$

现要计算其 Nichols 曲线(Ts=0.05)，则可输入

```
num=[1.5];
den=[1  1.1  1.36  0.88  0.31];
ngrid('new');
dnichols(num,den,0.05);
title('Discrete Nichols Plot')
```

执行后得到如图 2.29 所示的结果曲线。

参见：nichols，ngrid，dbode，dnyquise，logspace

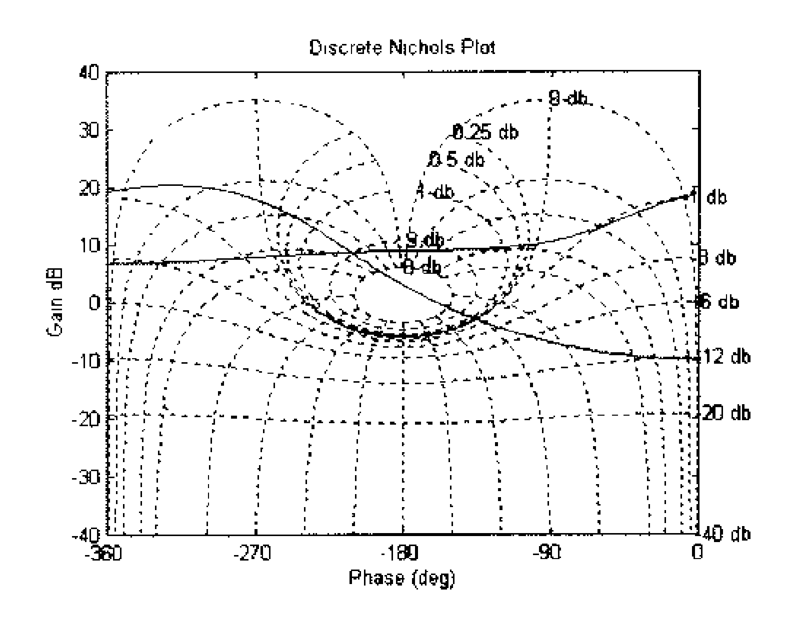


图 2.29　离散系统的 Nichols 曲线

7．ngrid

功能：绘制 Nichols(尼柯尔斯)曲线网格。

格式：

```
ngrid
ngrid('new')
```

说明：

ngrid 函数可给 Nichols 曲线图加上网格线，Nichols 曲线将复数 h/(1+h)与 h 联系在一起，当 h 为 SISO 系统的开环频率响应时，则 h/(1+h)为系统的闭环频率响应。

ngrid 函数可画出幅值 mag(h/(1+h))和相位 angle(h/(1+h))为常数的网格，幅值取－40dB～40dB，相位取－360°～0°。

ngrid('new')可在绘制网络前清除原图，然后再设置成 hold on，这样后续 Nichols 命令可以与网格绘制在一起，如输入

```
ngrid('new');
nichols(num,den);
```

有关示例见 nichols 函数。

参见：nichols，dnichols

8．sigma

功能：求连续状态空间系统的奇异值 Bode 图。

格式：

```
[sv,w]=sigma(a,b,c,d)
```

[sv,w]=sigma(a,b,c,d, 'inv')

[sv,w]=sigma(a,b,c,d,w)

[sv,w]=sigma(a,b,c,d,w, 'inv')

说明：

sigma 函数可计算出复矩阵 $c(j\omega I-a)^{-1}b+d$ 的奇异值，这种奇异值是 MIMO 系统 Bode 幅值响应的扩展。在不带输出变量引用时，sigma 函数可在当前图形窗口中绘出奇异值 Bode 图。

sigma(a,b,c,d)可求出作为频率函数的复矩阵

$$g(\omega)=c(j\omega I-a)^{-1}b+d$$

的奇异值 Bode 图，频率范围由函数自动选取。

对方阵系统，sigma(a,b,c,d, 'inv')可得到复矩阵逆

$$g^{-1}(\omega)=[c(j\omega I-a)^{-1}b+d]^{-1}$$

的奇异值 Bode 图。

sigma(a,b,c,d,w)或 sigma(a,b,c,d,w, 'inv')可利用指定的频率矢量来绘制奇异值 Bode 图。

当带输出变量引用函数时，可得到奇异值 Bode 图的数据，sv 为奇异值，w 为相应的频率值。

参见：dsigma，bode，dbode，logspace

9. dsigma

功能：求离散空间状态系统的奇异值 Bode 图。

格式：

[sv,w]=dsigma(a,b,c,d,Ts)

[sv,w]=dsigma(a,b,c,d,Ts, 'inv')

[sv,w]=dsigma(a,b,c,d,Ts,w)

[sv,w]=dsigma(a,b,c,d,Ts,w, 'inv')

说明：

dsigma 函数可计算出复矩阵 $c(e^{j\omega T}I-a)^{-1}b+d$ 的奇异值，这种奇异值是 MIMO 系统 Bode 幅值响应的扩展。在不带输出变量引用时，sigma 函数可在当前窗口中绘出奇异值 Bode 图。

dsigma(a,b,c,d,Ts)可求出作为频率函数的复矩阵

$$g(\omega)=c(e^{j\omega T}I-a)^{-1}b+d$$

的奇异值 Bode 图，频率范围由函数自动选取，Ts 为内部取样时间。

对方阵系统，dsigma(a,b,c,d,Ts, 'inv')可得到复矩阵逆

$$g^{-1}(\omega)=[c(e^{j\omega T}I-a)^{-1}b+d]^{-1}$$

的奇异值 Bode 图。

dsigma(a,b,c,d,Ts,w)或 dsigma(a,b,c,d,Ts,w, 'inv')，可利用指定的频率矢量来绘制奇异值 Bode 图。

当带输出变量引用函数时，可得到奇异值 Bode 图的数据，sv 为奇异值，w 为相应的频率值。

参见：sigma，bode，dbode，logspace

10. freqs

功能：模拟滤波器的频率响应。

格式：

```
h=freqs(b, a, w)
[h, w]=freqs(b, a)
[h, w]=freqs(b, a, n)
freqs(b, a)
```

说明：

freqs 用于计算由矢量 a 和 b 构成的模拟滤波器

$$H(s)=\frac{B(s)}{A(s)}=\frac{b(1)s^{n_b}+b(2)s^{(n_b-1)}+\cdots+b(n_b+1)}{s^{n_a}+a(2)s^{(n_a-1)}+\cdots+a(n_a+1)}$$

的复频响应 $H(j\omega)$。

h=freqs(b，a，w)用于计算模拟滤波器的复频响应，其中实矢量 w 用于指定频率值，即 freqs 沿虚轴计算频率响应。

[h，w]=freqs(b，a)自动设定 200 个频率点来计算频率响应，这 200 个频率值记录在 w 中。

[h，w]=freqs(b，a，n)设定 n 个频率点计算频率响应。

不带输出变量的 freqs 函数，将在当前图形窗口中绘制出幅频和相频曲线。

例 有一模拟滤波器，其传递函数设为

$$H(s)=\frac{0.2s^2+0.3s+1}{s^2+0.4s+1}$$

现可通过 freqs 函数可得到它的幅频特性和相频特性，其程序为

```
a=[1 0.4 1];
b=[0.2 0.3 1];
w=logspace(-1, 1);
h=freqs(b, a, w)
```

参见：freqz

11. freqz

功能：数字滤波器的频率响应。

格式：

```
[h, w]=freqz(b, a, n)
[h, f]=freqz(b, a, n, Fs)
[h, w]=freqz(b, a, n, 'whole')
[h, f]=freqz(b, a, n, 'whole', Fs)
```

```
h=freqz(b, a, w)
h=freqz(b, a, f, Fs)
freqz(b, a)
```

说明：

freqz 用于计算由矢量 a 和 b 构成的数字滤波器

$$H(z)=\frac{B(z)}{A(z)}=\frac{b(1)+b(2)z^{-1}+\cdots+b(n_b+1)z^{-n_b}}{1+a(2)z^{-1}+\cdots+a(n_a+1)z^{-n_a}}$$

的复频响应 H(jω)。

[h, w]=freqz(b, a, n)可得到数字滤波器的 n 点的复频响应，这 n 个点均匀地分布在上半单位圆(即 0～π)，并将这 n 点频率记录在 w 中，相应的频率响应记录在 h 中。至于 n 值的选择没有太多的限制，只要 n>0 的整数，但最好能选取 2 的幂次方，这样就可采用 FFT 算法进行快速计算。如果缺省，则 n=512。

[h, f]=freqz(b, a, n, Fs)允许指定采样终止频率 Fs(以 Hz 为单位)，也即在 0～Fs/2 频率范围内选取 n 个频率点(记录在 f 中)，并计算相应的频率响应 h。

[h, w]=freqz(b, a, n, 'whole')表示在 0～2π 之间均匀选取 n 个点计算频率响应。[h, f]=freqz(b, a, n, 'whole', Fs)则在 0～Fs 之间均匀选取 n 个点计算频率响应。

h=freqz(b, a, w)计算在矢量 w 中指定的频率处的频率响应，但必须注意，指定的频率必须介于 0～2π 之间。

h=freqz(b, a, f, Fs)计算在矢量 f 中指定的频率处的频率响应，但指定频率必须介于 0 ～Fs 之间。

不带输出变量的 freqz 函数可在当前图形窗口中绘制出幅频和相频特性曲线。

例　对一数字滤波器

$$H(z)=\frac{0.2+0.3z^{-1}+z^{-2}}{1+0.4z^{-1}+z^{-2}}$$

则可通过下列程序得到滤波器的幅频和相频特性

```
b=[0.2  0.3  1];
a=[1  0.4  1];
h=freqz(b, a, 128);
```

参见：freqs

12. margin

功能：求增益和相位裕度。

格式：

```
[gm,pm,wcp,wcg]=margin(mag,phase,w)
[gm,pm,wcp,wcg]=margin(num,den)
[gm,pm,wcp,wcg]=margin(a,b,c,d)
```

说明：

margin 函数可从频率响应数据中计算出增益、相位裕度以及有关的交叉频率。增益和相位裕度是针对开环 SISO 系统而言的，它指示出当系统闭环时的相对稳定性。当不带输

出变量引用时，margin 可在当前图形窗口中绘制出裕度的 Bode 图。

增益裕度是在相位为－180°处使环路增益为 1 的增益量，如在－180°相频处的增益为 g，则增益裕度为 1/g。类似地，相位裕度是当环路增益为 1.0 时，相应的相角与 180°之间的偏差。

margin(mag,phase,w)可得到增益和相位裕度，并绘制出 Bode 图，其中 mag，phase 和 w 为由 bode 或 dbode 得到的增益、相位裕度及其频率值。

margin(num,den)可计算出连续系统 g(s)＝num(s)/den(s)表示的增益和相位裕度，类似地，margin(a,b,c,d)可计算出以连续状态空间系统(a,b,c,d)表示的系统增益和相位裕度，这种格式只适用于连续系统；而对离散系统，可用 dbode 计算频率响应，然后调用 margin 函数

```
[mag,phase,w]=dbode(a,b,c,d,Ts);
margin(mag,phase,w)
```

[gm,pm,wcp,wcg]＝margin(mag,phase,w)可得到增益和相位裕度及相应的频率 wcg、wcp，而不直接绘制出 Bode 图曲线。

例 margin 函数通常放在 bode 函数之后，先由 bode 函数得到增益和相位裕度，然后由 margin 函数绘制出增益和相位裕度的 Bode 图，如输入

```
[a,b,c,d]=ord2(1, 0.2);
[mag,phase,w]=bode(a,b,c,d,1);
margin(mag,phase,w)
```

执行后得到如图 2.30 所示的曲线。

参见：logspace，bode，dbode，nichols，dnichols，nyquist

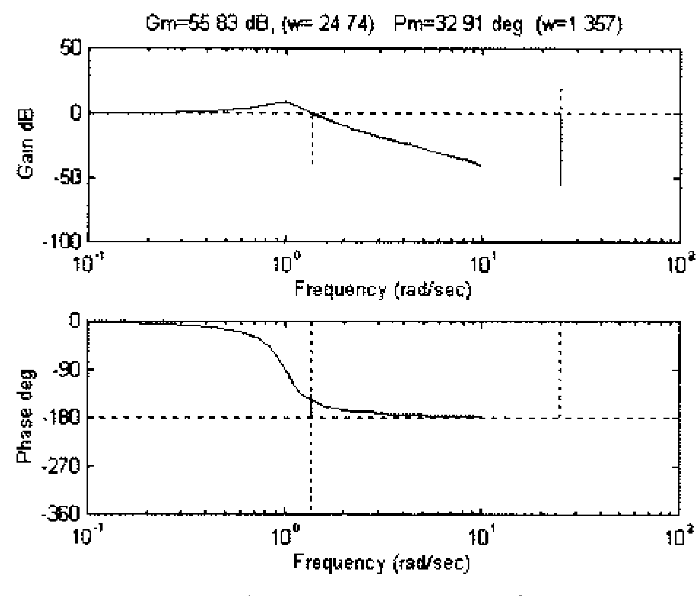


图 2.30 增益和相位裕度

13. ltifr

功能：求线性时不变频率响应。

格式：

ltifr(a,b,s)

说明：

ltifr 是涉及线性时不变状态空间系统的频率响应的计算核。它由 freqresp 函数调用，而 freqresp 函数为所有的频率响应函数引用。ltifr 函数本身一般不直接使用。

g=ltifr(a,b,s)可计算单输入系统

$$g(s) = (sI - a)^{-1}b$$

的频率响应。其中 s 用于指定复频率范围，对连续系统的 Bode 响应，s 取虚轴；对离散系统的 Bode 响应，s 取单位圆。

参见：bode，dbode，nyquist dnyquist，nichols，dnichols，ltitr，hess

2.9 根 轨 迹

1．pzmap

功能：绘制系统的零极点图。

格式：

```
[p,z]=pzmap(a,b,c,d)
[p,z]=pzmap(num,den)
[p,z]=pzmap(p,z)
```

说明：

pzmap 函数可绘出 LTI 系统的零极点图，对 SISO 系统而言，pzmap 函数可绘出传递函数的零极点；对 MIMO 系统而言，pzmap 可绘制出系统的特征矢量和传递零点。当不带输出变量引用时，pzmap 可在当前图形窗口中绘制出系统的零极点图。

pzmap(a,b,c,d)可在复平面内绘制出状态空间系统的零极点，对于 MIMO 系统，可绘出所有输入到输出间的传递零点，在图中，极点用“×”表示，零点用“o”表示。

pzmap(num,den)可在复平面内绘出以传递函数表示的系统零极点。

pzmap(p,z)可在复平面内绘制零极点图，其中列矢量 p 为极点位置，列矢量 z 为零点位置。这个命令用于直接绘制给定的零极点。

当带有输出变量引用函数时，可得到零极点位置，如需要可通过 pzmap(p,z)绘制出零极点图。

例 有连续系统

$$H(s) = \frac{2s^2 + 5s + 1}{s^2 + 2s + 3}$$

现要求绘制其零极点图，可输入

```
num=[2  5  1];
den=[1  2  3];
pzmap(num,den);
title('Pole-Zero Map')
```

执行后得到如图 2.31 所示的零极点图。

参见：eig，roots，rlocus，tzero，sgrid，zgrid

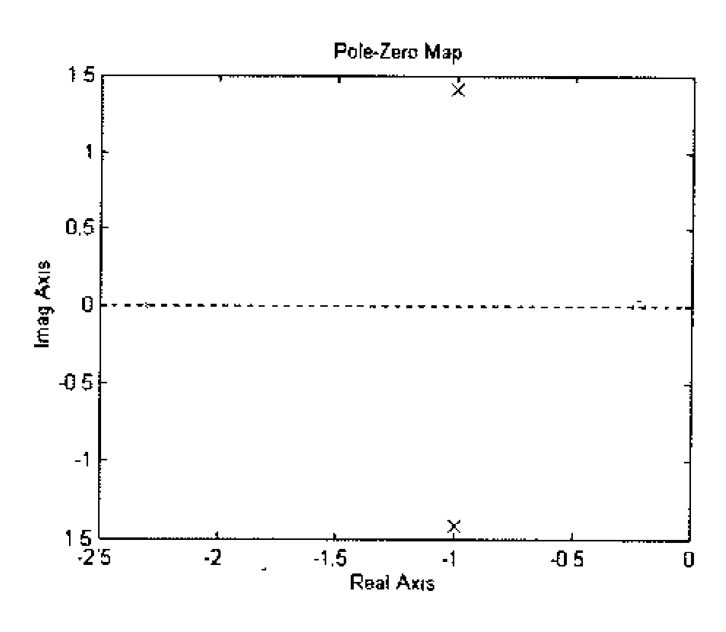


图 2.31 系统的零极点图

2. rlocus

功能：求系统根轨迹。

格式：

```
[r,k]=rlocus(num,den)
[r,k]=rlocus(num,den,k)
[r,k]=rlocus(a,b,c,d)
[r,k]=rlocus(a,b,c,d,k)
```

说明：

rlocus 函数可计算出 SISO 系统的 Evans(伊文斯)根轨迹。根轨迹可用于研究改变反馈增益对系统极点分布的影响，从而提供系统时域和频域响应的分析。给定传递函数 g(s)，反馈补偿为 k * f(s)的受控系统，其闭环传递函数为

$$h(s) = \frac{g(s)}{1 + kg(s)f(s)} = \frac{g(s)}{q(s)}$$

在不带输出变量引用函数时，rlocus 可在当前图形窗口中绘制出系统的根轨迹图。rlocus 函数既适用于连续时间系统，也适用于离散时间系统。

rlocus(num,den)可绘出 q(s)=1+k * num(s)/den(s)=0 的根轨迹，增益 k 是自动选取的。

rlocus(a,b,c,d)可绘出以连续或离散时间 SISO 状态空间系统(a,b,c,d)的根轨迹，增益 k 也是自动选取的。

rlocus(num,den,k)或 rlocus(a,b,c,d,k)可利用指定的增益 k 来绘制系统的根轨迹。

当带有输出变量引用函数时，可得到复根轨迹的位置矩阵 r 及相应的增益矢量 k。利用 plot(r, 'x')可绘制出根轨迹。

例 有系统

$$H(s)=\frac{2s^2+5s+1}{s^2+2s+3}$$

现要绘制出其根轨迹，可输入

```
num=[2  5  1];
den=[1  2  3];
rlocus(num,den);
title('Root Locus')
```

执行后得到如图 2.32 所示的系统根轨迹。

参见：rlocfind，pzmap，place，lqr

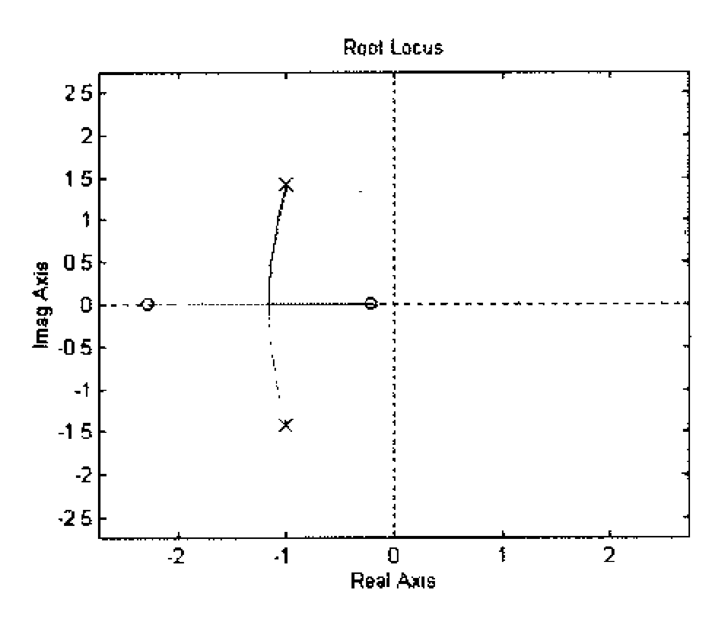


图 2.32　系统的根轨迹

3. rlocfind

功能：计算给定一组根的根轨迹增益。

格式：

```
[k,poles]=rlocfind(a,b,c,d)
[k,poles]=rlocfind(a,b,c,d,p)
[k,poles]=rlocfind(num,den)
[k,poles]=rlocfind(num,den,p)
```

说明：

rlocfind 函数可计算出与根轨迹上极点相对应的根轨迹增益。rlocfind 既适用于连续时间系统，也适用于离散时间系统。

[k,poles]=rlocfind(a,b,c,d)可在图形窗口根轨迹图中显示出十字光标，当用户选择其中一点时，其相应的增益由 k 记录，与增益相关的所有极点记录在 poles 中。

[k,poles]=rlocfind(num,den)可在以传递函数 g=unm/den 表示的系统根轨迹图上选

取轨迹。

[k,poles]=rlocfind(a,b,c,d,p)或[k,poles]=rlocfind(num,den,p)可指定要得到增益的根矢量 p。

参见：rlocus

4. sgrid

功能：在连续系统根轨迹图和零极点图中绘制出阻尼系数和自然频率栅格。

格式：

```
sgrid
sgrid( 'new')
sgrid(z,Wn)
sgrid(z,Wn, 'new')
```

说明：

sgrid 函数可在连续系统的根轨迹或零极点图上绘制出栅格线，栅格线由等阻尼系数和等自然频率线构成，阻尼系数线以步长 0.1 从 $\xi=0$ 到 $\xi=1$ 绘出。

sgrid('new')函数先清除图形屏幕，然后绘制出栅格线，并设置成 hold on，使后续绘图命令能绘制在栅格上。典型用法如

```
sgrid('new')
rlocus(num,den)或 pzmap(num,den)
```

sgrid(z,Wn)可指定阻尼系数 z 和自然频率 Wn。

sgrid(z,Wn, 'new')可指定阻尼系数 z 和自然频率 Wn，并且在绘制栅格线之前清除图形窗口。

例 有系统

$$H(s)=\frac{2s^2+5s+1}{s^2+2s+3}$$

可输入

```
unm=[2  5  1];
den=[1  2  3];
rlocus(num, den)
sgrid
title( 'Root Locus')
```

执行后得到如图 2.33 所示的根轨迹。

参见：rlocus, pzmap, rlocfind

5. zgrid

功能：在离散系统根轨迹和零极点图中绘制出阻尼系数和自然频率栅格线。

格式：

```
zgrid
```

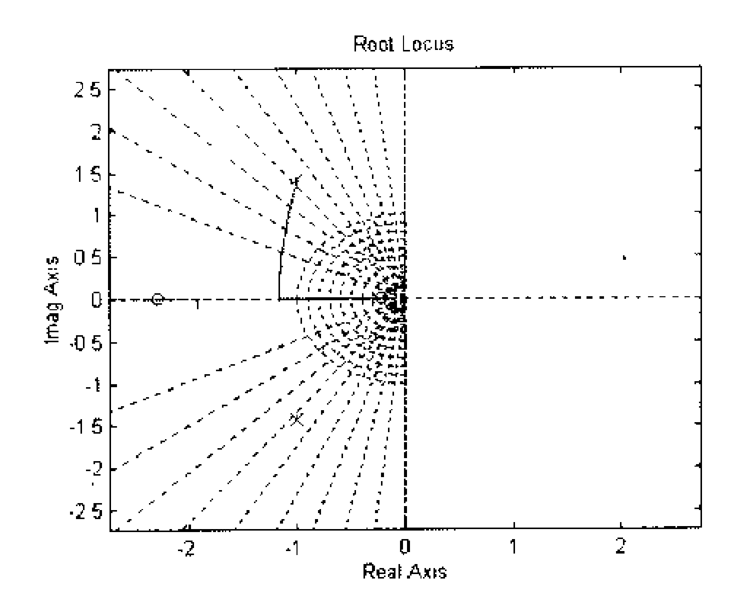


图 2.33　带栅格线的系统根轨迹

```
zgrid( 'new')
zgrid(z,Wn)
zgrid(z,Wn, 'new')
```

说明：

zgrid 函数可在离散系统的根轨迹图或零极点图上绘制出栅格线，栅格线由等阻尼系数和自然频率线构成，阻尼系数线以步长 0.1 从 $\xi=0$ 到 $\xi=1$ 绘出，自然频率线以步长 $\pi/10$ 从 0 到 π 绘出。

zgrid('new')函数先清除图形屏幕，然后绘制出栅格线，并设置成 hold on，使后续绘图命令能绘制在栅格上。典型用法如

```
zgrid( 'new')
rlocus(new,den) 或 pzmap(num,den)
```

zgrid(z,Wn)可指定阻尼系数 z 和自然频率 Wn。非归一化频率的等频率线可采用

```
zgrid(z,Wn/Ts)
```

绘制，其中 Ts 为采样时间。

zgrid(z,Wn, 'new')可指定阻尼系数 z 和自然频率 Wn，并在绘制栅格线之前清除图形屏幕窗口。

zgrid([ ],[ ])可绘制出单位圆。

例　有系统

$$H(z)=\frac{2z^2-3.4z+1.5}{z^2-1.6z+0.8}$$

可输入

```
num=[2  -3.4  1.5];
den=[1  -1.6  0.8];
axis( 'square')
zgrid( 'new')
```

```
rlocus(num,den);
title('Root Locus')
```

执行后得到如图 2.34 所示的根轨迹。

参见：rlocus，pzmap，rlocfind

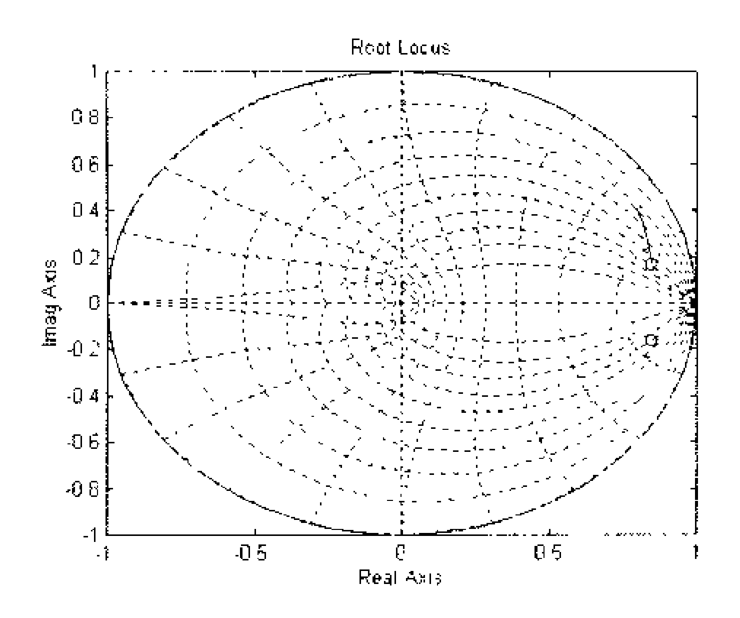


图 2.34　带栅格线的系统根轨迹

2.10　估计器/调节器设计

1. lqe，lqe2，lqew

功能：连续系统线性二次型估计器设计。

格式：

```
[l,p,e]=lqe(a,g,c,Q,R)
[l,p,e]=lqe(a,g,c,Q,R,N)
[l,p,e]=lqe2(a,g,c,…)
[l,p,e]=lqew(a,g,c,j,Q,R)
```

说明：

lqe，lqe2 和 lqew 函数可求解连续时间系统的线性二次型估计器问题及其相关的 Riccati 方程。

对于连续时间系统的状态方程和测量方程为

$$\dot{x} = ax + bu + gw$$
$$y = cx + du + v$$

其状态噪声的协方差为

$$E[w] = E[v] = 0, \quad E[ww^T] = Q,$$
$$E[vv^T] = R, \qquad E[wv^T] = 0$$

l=lqe(a,g,c,Q,R)可得到增益矩阵l，使连续稳态的卡尔曼滤波器

$$\dot{x} = ax + bu + l(y - cx - du)$$

产生x的LQG最佳估计。

[l,p,e]=lqe(a,g,c,Q,R)还得到了Riccati方程的解p和估计器的闭环特征值e=eig(a-l*c)。

[l,p,e]=lqe(a,g,c,Q,R,N)可计算当状态噪声和测量噪声相关时的卡尔曼矩阵，即

$$E[wv^T] = N$$

[l,p,e]=lqew(a,g,c,j,Q,R)可针对以下系统

$$\dot{x} = ax + bu + gw$$

$$y = cx + du + jw + v$$

计算卡尔曼增益矩阵，其状态噪声和测量噪声不相关。注意这与lqe函数适用的系统有所不同。

[l,p]=lqe2(a,g,c,Q,R)或[l,p]=lqe2(a,g,c,Q,R,N)与lqe类似，只是算法中采用了Schur方法，这可使系统具有更强的鲁棒性。

参见：lqr，dlqe，dlqew

2. dlqe，dlqew

功能：离散系统线性二次型估计器设计。

格式：

[l,m,p,e]=dlqe(a,g,c,Q,R)

[l,m,p,e]=dlqe(a,g,c,Q,R,N)

[l,m,p,e]=dlqew(a,g,c,j,Q,R)

说明：

dlqe和dlqew函数可求解离散时间系统的线性二次型估计器问题及其相关的Riccati方程。

对于离散系统的状态方程和测量方程

$$x[n+1] = ax[n] + bu[n] + gw[n]$$

$$y[n] = cx[n] + du[n] + v[n]$$

其状态噪声和测量噪声的协方差为

$$E[w] = E[v] = 0, \quad E[ww^T] = Q,$$

$$E[vv^T] = R, \quad E[wv^T] = 0$$

l=dlqe(a,g,c,Q,R)可得到增益矩阵l，使离散稳态的卡尔曼滤波器

$$\hat{x}[n] = \bar{x}[n] + l(y[n] - c\bar{x}[n] - du[n])$$

产生x的LQG最佳估计。

[l,m,p,e]=dlqe(a,g,c,Q,R)还得到Riccati方程的解m，估计误差方差p=E[(x̂-x)(x̂-x)']及估计器闭环特征值e=eig(a-a*l*c)。

[l,m,p,e]=dlqe(a,g,c,Q,R,N)可计算当状态噪声和测量噪声相关时的卡尔曼增益矩阵，即

$$E[wv^T] = N$$

[l,m,p,e]=dlqew(a,g,c,j,Q,R)可针对以下系统

$$x[n+1] = ax[n] + bu[n] + gw[n]$$
$$y[n] = cx[n] + du[n] + jw[n] + v[n]$$

计算卡尔曼矩阵，其状态噪声和测量噪声不相关。注意，这与 dlqe 函数适用的系统有所不同。

参见：lqe，lqew，lqed，destim

3. lqed

功能：根据连续代价函数设计离散估计器。

格式：

```
[l,m,p,e]=lqed(a,g,c,Q,R,Ts)
```

说明：

lqed 函数设计的离散估计器与利用 lqe 函数设计的连续估计器具有相似的响应特性，这样在得到满意的连续估计器设计之后，可利用这一函数为数字实现设计出离散估计器。

l=lqed(a,g,c,Q,R,Ts)可得到离散增益矩阵 l，使从连续估计器到离散估计器的等效误差最小，其中连续系统的状态噪声和测量噪声的协方差为

$$E[w] = E[v] = 0,\quad E[ww^T] = Q,$$
$$E[vv^T] = R,\qquad E[wv^T] = 0$$

Ts 为采样时间。得到的估计器观测更新律为

$$\hat{x}[n] = \bar{x}[n] + l(y[n] - c_d\bar{x}[n] - d_d u[n])$$

状态更新律为

$$\bar{x}[n+1] = a_d\hat{x}[n] + b_d u[n]$$

其中离散系统(a_d,b_d,c_d,d_d)可利用 c2d 函数得到。

[l,m,p,e]=lqed(a,g,c,Q,R,Ts)可得到离散卡尔曼增益矩阵 l、离散 Riccati 方程解 m、测量更新后的估计协方差 p 及离散闭环特征值 e

$$e=eig(ad-ad*l*cd)$$
$$m=E[(x-\bar{x})(x-\bar{x})^T]$$
$$p=E[(x-\hat{x})(x-\hat{x})^T]$$

参见：lqr，dlqe，dlqr

4. lqr，lqr2，lqry

功能：连续系统的线性二次型调节器设计。

格式：

```
[k,s,e]=lqr(a,b,Q,R)
[k,s,e]=lqr(a,b,Q,R,N)
[k,s]=lqr2(a,b,…)
[k,s,e]=lqry(a,b,c,d,Q,R)
```

说明：

lqr、lqr2 和 lqry 函数可求解连续系统线性二次型调节稳定器问题及其相关的 Riccati 方程。

k=lqr(a,b,Q,R)可计算出最佳反馈增益矩阵 k，采用反馈律

$$u = -kx$$

使代价函数

$$J = \int (x^T Q x + u^T R u)\, dt$$

最小，当然这要受状态方程的约束。

[k,s,e]=lqr(a,b,Q,R)还可得到使 Riccati 方程

$$0 = sa + a^T s - sbR^{-1}b^T s + Q$$

成立的正定阵 s 及系统的特征值 e

e=eig(a−b * k)

[k,s,e]=lqr(a,b,Q,R,N)可求得最佳反馈增益矩阵，使代价函数

$$J = \int (x^T Q x + 2u^T N x + u^T R u)\, dt$$

最小。

[k,s,e]=lqry(a,b,c,d,Q,R)可求得最佳反馈增益矩阵，使代价函数

$$J = \int (y^T Q y + u^T R u)\, dt$$

最小。

[k,s]=lqr2(a,b,Q,R)或[k]=lqr2(a,b,Q,R,N)与 lqr 函数类似，只是其算法采用了 Schar 方法，因此具有更强的鲁棒性。

参见：lqe，dlqr，dlqry

5. dlqr，dlqry

功能：离散系统的线性二次型调节器设计。

格式：

[k,s,e]=dlqr(a,b,Q,R)

[k,s,e]=dlqr(a,b,Q,R,N)

[k,s,e]=dlqry(a,b,c,d,Q,R)

说明：

dlqr 和 dlqry 函数可求解离散时间系统的线性二次型调节器问题及其相关的 Riccati 方程。

k=dlqr(a,b,Q,R)可计算出最佳反馈增益矩阵 k，使得采用反馈律

$$u = -kx$$

使代价函数

$$J = \sum (x^T Q x + u^T R u)$$

最小，当然这要受状态方程的约束。

[k,s,e]=dlqr(a,b,Q,R)，还可得到使 Riccati 方程

$$0 = s - a^T sa + a^T sb(R + b^T sb)^{-1} bs^T a - Q$$

成立的正定阵 s，也得到了闭环系统的特征值 e

e=eig(a-b * k)

[k,s,e]=dlqr(a,b,Q,R,N)，可求得最佳反馈增益矩阵，使代价函数

$$J = \sum (x^T Qx + 2u^T Nx + u^T Ru)$$

最小。

[k,s,e]=dlqry(a,b,c,d,Q,R)，可求得最佳反馈增益矩阵，使代价函数

$$J = \sum (y^T Qy + u^T Ru)$$

最小。

参见：lqr，lqry，lqrd，dreg

6. lqrd

功能：根据连续代价函数设计离散调节器。

格式：

[k,s,e]=lqrd(a,b,Q,R,Ts)

[k,s,e]=lqrd(a,b,Q,R,N,Ts)

说明：

lqrd 函数设计的离散全状态反馈控制器与利用 lqr 函数设计的连续全状态反馈控制器具有相似的响应特性，这样在得到满意的连续增益矩阵设计之后，可利用这一函数为数字实现设计出全状态反馈增益矩阵。

k=lqrd(a,b,Q,R,Ts)，可得到离散反馈增益矩阵 k，使连续代价函数

$$J = \int_0^{Ts} (x^T Qx + u^T Ru)\,dt$$

的等效离散代价函数为最小，其中 Ts 为采样时间。

[k,s,e]=lqrd(a,b,Q,R,Ts)可求得离散反馈增益矩阵 k，离散 Riccati 方程的解 s，及离散闭环系统的特征值 e

e=eig(ad-bd * k)

[k,s,e]=lqrd(a,b,Q,R,N,Ts)可计算出离散反馈增益矩阵，使代价函数

$$J = \int_0^{Ts} (x^T Qx + 2u^T Nx + u^T Ru)\,dt$$

最小。

参见：lqe，dlqe，dlqr

7. place，acker

功能：极点配置增益选择。

格式：

k=place(a,b,p)

k=acker(a,b,p)

说明:

place 和 acker 函数用于极点配置增益选择。

k=acker(a,b,p)，可利用 Ackermann 公式计算反馈增益矩阵 k，使采用全反馈 u=−kx的单输入系统

$$\dot{x} = ax + bu$$

具有指定的闭环极点 p，即

p=eig(a−b*k)

k=place(a,b,p)，可在假定对象所有输入为控制输入时计算出反馈增益矩阵，以便获得期望的闭环极点。place 函数适用于多输入系统。

place 和 acker 函数也可以用来计算估计器增益，这时应采用

l=place(a′, c′, p)

l=acker(a′, c′, p)

参见: lqr, lqe, rlocus

第3章

控制系统分析与设计

在第1章控制系统基本理论和第2章控制系统工具箱函数的基础上，本章利用MATLAB及其工具箱函数来解决控制系统的分析与设计问题。

本章的每一节均与第1章的节相对应，给出相关问题的分析与设计示例。在每个例子中，均给出MATLAB程序，它可直接在MATLAB V4.21c中直接执行，并得到相应的结果及图形。

3.1 控制系统模型

例3.1 设系统的零极点增益模型为

$$H(s)=\frac{6(s+3)}{(s+1)(s+2)(s+5)}$$

求系统的传递函数及状态空间模型。

解 在MATLAB中，可直接利用zp2tf和zp2ss函数求出，MATLAB程序为ex3001.m：

```
% Example 3.1
%
k=6;
z=[-3];
p=[-1,-2,-5];
[num,den]=zp2tf(z,p,k)
[a,b,c,d]=zp2ss(z,p,k)
```

执行后得

```
num=
      0  0  6  18
den=
```

```
        1    8   17   10
a=
    -1.0000         0         0
     2.0000   -7.0000   -3.1623
          0    3.1623         0
b=
     1
     1
     0
c=
     0    0    1.8974
d=
     0
```

因此，系统的传递函数模型为

$$H(s) = \frac{6s + 18}{s^3 + 8s^2 + 17s + 10}$$

系统的状态空间模型为

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} -1.0000 & 0 & 0 \\ 2.0000 & -7.0000 & -3.1623 \\ 0 & 3.1623 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} u(t) \\ y(t) = \begin{bmatrix} 0 & 0 & 1.8974 \end{bmatrix} x(t) \end{cases}$$

例 3.2 给定离散系统状态空间方程

$$\begin{cases} x(k+1) = \begin{bmatrix} -2.8 & -1.4 & 0 & 0 \\ 1.4 & 0 & 0 & 0 \\ -1.8 & -0.3 & -1.4 & -0.6 \\ 0 & 0 & 0.6 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} u(k) \\ y(k) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} x(k) \end{cases}$$

求传递函数模型和零极点模型，并判断其稳定性。

解 MATLAB 程序为 ex3002.m：

```
% Example 3.2
%
a=[-2.8 -1.4 0 0; 1.4 0 0 0; -1.8 -0.3 -1.4 -0.6;0 0 0.6 0];
b=[1;0;1;0];
c=[0,0,0,1];d[0];
[unm,den]=ss2tf(a,b,c,d)
[z,p,k]=ss2zp(a,b,c,d)
pzmap(p,z)
title( 'Pole-Zero Map')
```

执行后得

```
num=
      0   0   0.6   0.6   0.9240
den=
     1   4.2   6.24   3.752   0.7056
z=
   -0.5+1.1358i
   -0.5-1.1358i
p=
   -1.4
   -1.4
   -1.0606
   -0.3394
k=
     0.6
```

并且得到了如图 3.1 所示的系统零极点图。

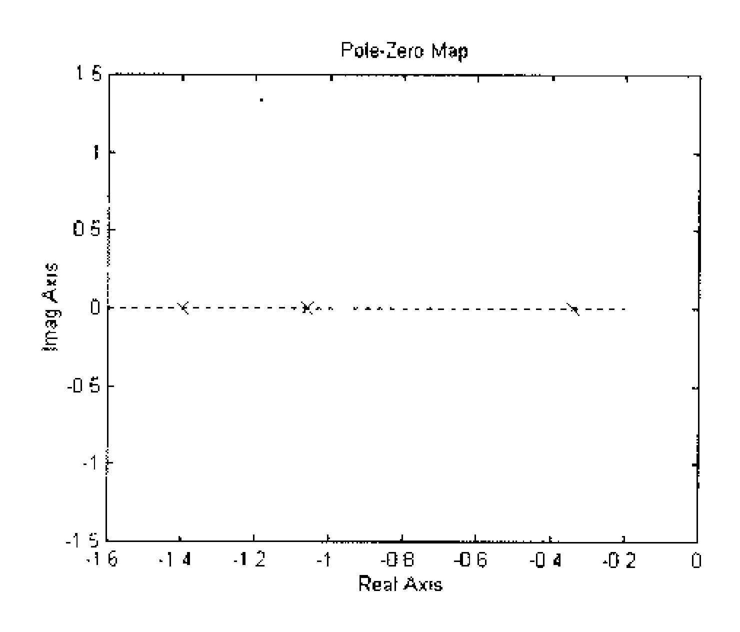


图 3.1 系统的零极点图

从计算结果可以得到，系统的传递函数和零极点模型为

$$H(s)=\frac{0.6s^2+0.6s+0.924}{s^4+4.2s^3+6.24s^2+3.752s+0.7056}$$

$$=0.6\frac{(s+0.5-1.1358i)(s+0.5+1.1358i)}{(s+1.4)^2(s+1.0606)(s+0.3394)}$$

从图 3.1 中可以看出，系统极点和零点均位于左半 s-平面，因此，系统是稳定的，而且是最小相位的。

例 3.3 已知两个系统

$$\begin{cases}\dot{x}_1 = \begin{bmatrix}0 & 1\\ -1 & -2\end{bmatrix}x_1 + \begin{bmatrix}0\\1\end{bmatrix}u_1\\ y_1 = [1 \quad 3]x_1 + u_1\end{cases}$$

$$\begin{cases}\dot{x}_2 = \begin{bmatrix}0 & 1\\ -1 & -3\end{bmatrix}x_2 + \begin{bmatrix}0\\1\end{bmatrix}u_2\\ y_2 = [1 \quad 4]\ x_2\end{cases}$$

求按串联、并联、单位负反馈、单位正反馈连接时的系统状态方程。

解 在 MATLAB 中，两个系统的连接可直接采用 series、parallel、cloop 等函数实现。MATLAB 程序为 ex3003.m：

```
% Example 3.3
%
a1=[0,1;-1,-2];
b1=[0;1]
c1=[1,3];d1=[1];
a2=[0,1;-1,-3];
b2=[0;1]
c2=[1,4];d2=[0];
[a,b,c,d]=series(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d]=parallel(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2)
[a,b,c,d]=feedback(a1,b1,c1,d1,a2,b2,c2,d2,+1)
```

执行后可得

串联连接：

```
a=
     0    1    0    0
    -1   -2    0    0
     0    0    0    1
     1    3   -1   -3
b=
   0
   1
   0
   1
c=
   0    0    1    4
d=
   0
```

并联连接：

```
a=
     0     1     0     0
    -1    -2     0     0
     0     0     0     1
     0     0    -1    -3
b=
     0
     1
     0
     1
c=
     1     3     1     4
d=
     1
```

负反馈连接：

```
a=
     0     1     0     0
    -1    -2    -1    -4
     0     0     0     1
     1     3    -2    -7
b=
     0
     1
     0
     1
c=
     1     3    -1    -4
d=
     1
```

正反馈连接：

```
a=
     0     1     0     0
    -1    -2     1     4
     0     0     0     1
     1     3     0     1
b=
     0
     1
```

```
      0
      1
   c=
      1    3    1    4
   d=
      1
```

例 3.4 时不变线性系统

$$\begin{cases}\dot{x} = \begin{bmatrix}-3 & 1\\ 1 & -3\end{bmatrix}x + \begin{bmatrix}1 & 1\\ 1 & 1\end{bmatrix}u\\ y = \begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}x\end{cases}$$

判别系统的能控性和能观性。

解 MATLAB 程序为 ex3004.m：

```
% Example 3.4
%
a=[-3,1;1,-3];
b=[1,1;1,1];
c=[1,1;1,-1];d=[0];
cam=ctrb(a,b);
rcam=rank(cam)
oam=obsv(a,c);
roam=rank(oam)
```

执行后得可控性矩阵和可观性矩阵的秩

```
rcam=
     1
roam=
     2
```

由于系统阶次为 2，因此这一系统为不可控但可观的系统。

例 3.5 线性系统

$$H(s) = \frac{s+\alpha}{s^3 + 10s^2 + 27s + 18}$$

当 α 分别取 $-1, 0, +1$ 时，判别系统的可控性和可观性，并求出相应的状态方程。

解：MATLAB 程序为 ex3005.m：

```
% Example 3.5
%
for alph=[-1:1]
  alph
  num=[1,alph];
  den=[1 10 27 18];
```

```
  [a,b,c,d]=tf2ss(num,den)
  cam=ctrb(a,b);
  rcam=rank(cam)
  oam=obsv(a,c);
  coam=rank(oam)
end
```

执行后得 α 的不同值时的状态矩阵(a,b,c,d)，并求出其可控性矩阵、可观性矩阵的秩

```
alph=
      -1
a=
   -10   -27   -18
     1     0     0
     0     1     0
b=
   1
   0
   0
c=
   0     1     -1
d=
   0
rcam=
      3
coam=
      3

alph=
      0
a=
   -10   -27   -18
     1     0     0
     0     1     0
b=
   1
   0
   0
c=
   0     1     0
d=
   0
```

```
rcam=
         3
coam=
         3

alph=
        1
a=
    -10   -27   -18
      1     0     0
      0     1     0
b=
    1
    0
    0
c=
    0    1    1
d=
    0
rcam=
         3
coam=
         2
```

由于系统阶次为 3，因此，当 $\alpha=-1,0$ 时，系统为完全可控和完全可观的；当 $\alpha=1$ 时，系统为可控而不可观的。

例 3.6 利用李亚普诺夫方程确定线性时不变系统

$$\begin{cases}\dot{x}_1 = - x_1 - 2x_2 \\ \dot{x}_2 = x_1 - 4x_2\end{cases}$$

的稳定性。

解 $a = \begin{bmatrix} -1 & -2 \\ 1 & -4 \end{bmatrix}$

令 q=I，求解 Lyapunov 方程

$$ap + pa^{T} = -q$$

然后确定 p 的正定性来证实系统的稳定性。

MATLAB 程序为 ex3006.m：

```
% Example 3.6
%
a=[-1,-2;1,-4];
q=[1,0;0,1];
```

```
p=lyap(a,q)
detp=det(p)
```

执行后得

```
p=
    0.4333    0.0333
    0.0333    0.1333
detp=
      0.0567
```

观察 p 阵，由于 det(p(1,1))=0.4333，det(p)=0.0567，因此 p 阵为正定阵，因此该系统是稳定的。

例 3.7 对例 3.1 中的连续模型，采用 MATLAB 提供的各种函数进行离散化处理，从而得稍有差异的状态方程。

解 MATLAB 程序为 ex3007.m：

```
% Example 3.7
%
% Continue system
disp( 'Continue System')
k=6;
z=[-3];
p=[-1,-2,-5];
[a,b,c,d]=zp2ss(z,p,k)
% Discrete system
t=0.1;
disp( 'Discrete System - - using c2d')
[a1,b1]=c2d(a,b,t)
disp( 'Discrete System - - using c2dm with zoh')
[a2,b2,c2,d2]=c2dm(a,b,c,d,t, 'zoh')
disp( 'Discrete System - - using c2dm with foh')
[a3,b3,c3,d3]=c2dm(a,b,c,d,t, 'foh')
disp( 'Discrete System - - using c2dm with Tustin')
[a4,b4,c4,d4]=c2dm(a,b,c,d,t, 'tustin')
```

执行后得各种变换方法下的离散化系统

```
Continue System
a=
   -1.0000         0         0
    2.0000   -7.0000   -3.1623
         0    3.1623         0
b=
    1
```

```
    1
    0
c=
         0       0       1.8974
d=
    0

Discrete System - - using c2d
a1=
     0.9048         0          0
     0.1338    0.4651    -0.2237
     0.0243    0.2237     0.9602
b1=
     0.0952
     0.0784
     0.0135

Discrete System - - using c2dm with zoh
a2=
     0.9048         0          0
     0.1338    0.4651    -0.2237
     0.0243    0.2237     0.9602
b2=
     0.0952
     0.0784
     0.0135
c2=
         0       0   1.8974
d2=
     0

Discrete System  - using c2dm with foh
a3=
     0.9048         0          0
     0.1338    0.4651    -0.2237
     0.0243    0.2237     0.9602
b3=
     0.0906
     0.0611
```

```
     0.0240
c3=
          0      0   1.8974
d3=
     0.0089

Discrete System - - using c2dm with Tustin
a4=
     0.9048        0          0
     0.1385   0.4545    -0.2300
     0.0219   0.2300     0.9636
b4=
     0.9524
     0.7965
     0.1259
c4=
     0.0021     0.0218     0.1863
d4=
     0.0119
```

3.2　控制系统的时域分析

例 3.8　典型二阶系统

$$H(s)=\frac{\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2}$$

其中 ω_n 为自然频率(无阻尼振荡频率)，ξ 为相对阻尼系数。试绘制出当 $\omega_n=6$，ξ 分别为 0.1，0.2，…，1.0，2.0 时的单位阶跃响应。

解　MATLAB 程序为 ex3008.m：

```
% Example 3.8
%
wn=6;
kosi=[0.1:0.1:1.0,2.0];
figure(1)
hold on
for kos=kosi
   num=wn.^2;
   den=[1,2*kos*wn, wn.^2];
   step(num,den)
```

```
end
title( 'Step Response')
hold off
```

执行后得如图 3.2 所示的单位阶跃响应曲线。

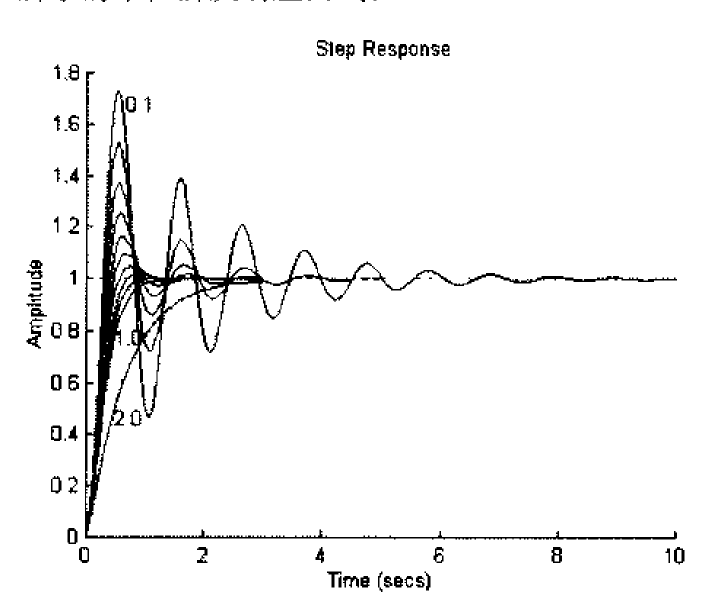


图 3.2 典型二阶系统的单位阶跃响应曲线

从图 3.2 中可以看出，在过阻尼和临界阻尼响应曲线中，临界阻尼响应具有最短的上升时间，响应速度最快；在欠阻尼($0<\xi<1$)响应曲线中，阻尼系数越小，超调量越大，上升时间越短，通常取 $\xi=0.4\sim0.8$ 为宜，这时超调量适度，调节时间较短。

例 3.9 对例 3.8 中的典型二阶系统，绘制出当 $\xi=0.7$，ω_n 取 2,4,6,8,10,12 时的单位阶跃响应。

解 MATLAB 程序为 ex3009.m：

```
% Example 3.9
%
w=[2:2:12];
kos=0.7;
figure(1)
hold on
for wn=w
  num=wn.^2;
  den=[1, 2*kos*wn, wn.^2];
  step(num,den)
end
title( 'Step Response')
hold off
```

执行后得如图 3.3 所示的单位阶跃响应曲线。

从图 3.3 中可以看出，ω_n 越大，响应速度越快。

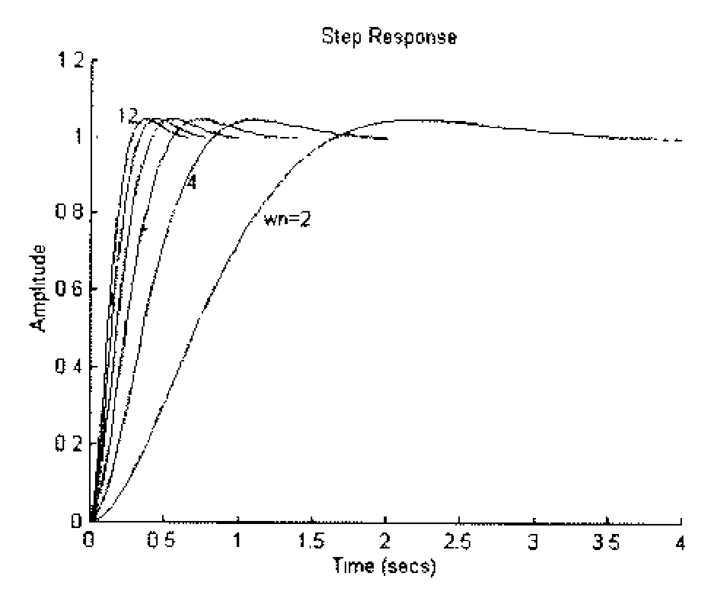


图 3.3　典型二阶系统的单位阶跃响应曲线

例 3.10　求三阶系统

$$H(s)=\frac{5(s^2+5s+6)}{s^3+6s^2+10s+8}$$

的单位阶跃响应。

解　MATLAB 程序为 ex3010.m：

```
% Example 3.10
%
num=[5 25 30];
den=[1 6 10 8];
step(num,den)
title('Step Response')
```

执行后得如图 3.4 所示的单位阶跃响应曲线。

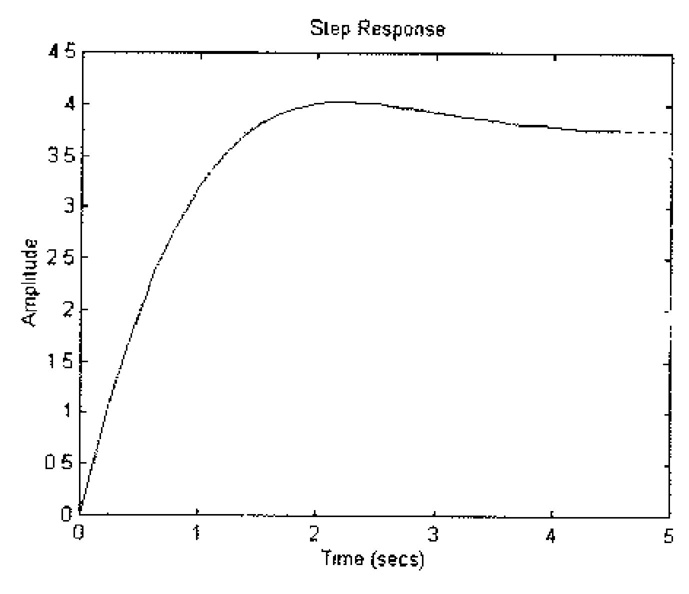


图 3.4　三阶系统的单位阶跃响应曲线

例 3.11 求典型二阶系统

$$H(s)=\frac{\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2}$$

当 $\xi=0.7$，$\omega_n=6$ 时的单位冲激响应。

解 MATLAB 程序为 ex3011.m：

```
% Example 3.11
%
wn=6;
kos=0.7
figure(1)
num=wn.^2;
den=[1, 2*kos*wn, wn.^2];
impulse(num,den)
title('Impulse Response')
```

执行后得如图 3.5 所示的单位冲激响应曲线。

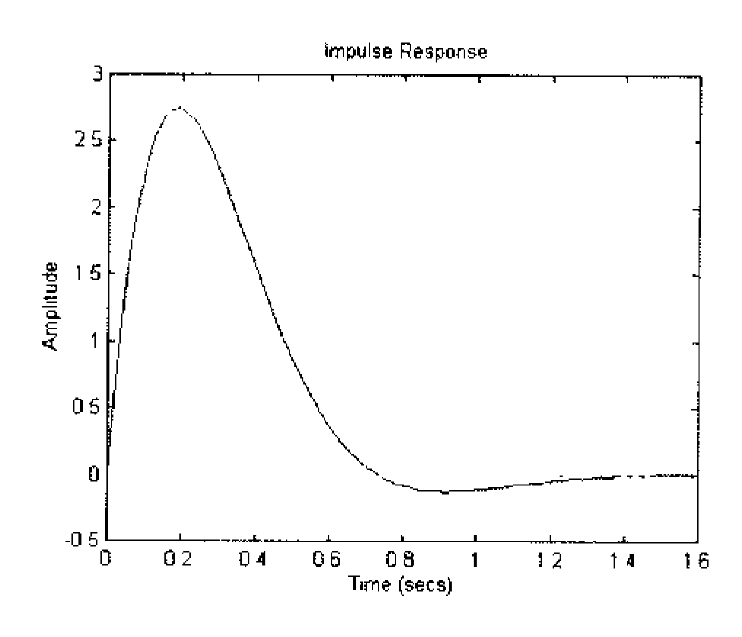


图 3.5 典型二阶系统的单位冲激响应曲线

例 3.12 有高阶系统

$$\begin{cases}\dot{x}=\begin{bmatrix}-1.6 & -0.9 & 0 & 0\\ 0.9 & 0 & 0 & 0\\ 0.4 & 0.5 & -5.0 & -2.45\\ 0 & 0 & 2.45 & 0\end{bmatrix}x+\begin{bmatrix}1\\0\\1\\0\end{bmatrix}u\\ y=\begin{bmatrix}1 & 1 & 1 & 1\end{bmatrix}x\end{cases}$$

求单位阶跃响应，单位冲激响应及零输入响应（设初始状态 $x_0=[1\ 1\ 1\ -1]^T$）。

解 MATLAB 程序为 ex3012.m：

```
% Example 3.12
%
```

```
a=[-1.6,-0.9,0,0; 0.9,0,0,0
   0.4,0.5,-5.0,-2.45; 0,0,2.45,0];
b=[1;0;1;0];
c=[1,1,1,1];
d=[0];
figure(1)
subplot(2,2,1)
step(a,b,c,d)
title( 'Step Response' )
subplot(2,2,2)
impulse(a,b,c,d)
title( 'Impulse Response' )
subplot(2,2,3)
x0=[1;1;1;-1];
initial(a,b,c,d,x0)
axis([0 6 -0.5 2.5])
title( 'Initial Response' )
subplot(2,2,4)
pzmap(a,b,c,d)
title( 'Pole-Zero Map' )
```

执行后得如图 3.6 所示的曲线，其中包含系统的单位阶跃响应、单位冲激响应、零输入响应，在右下角还给出了系统的零极点图，从中可以看出系统是稳定的。

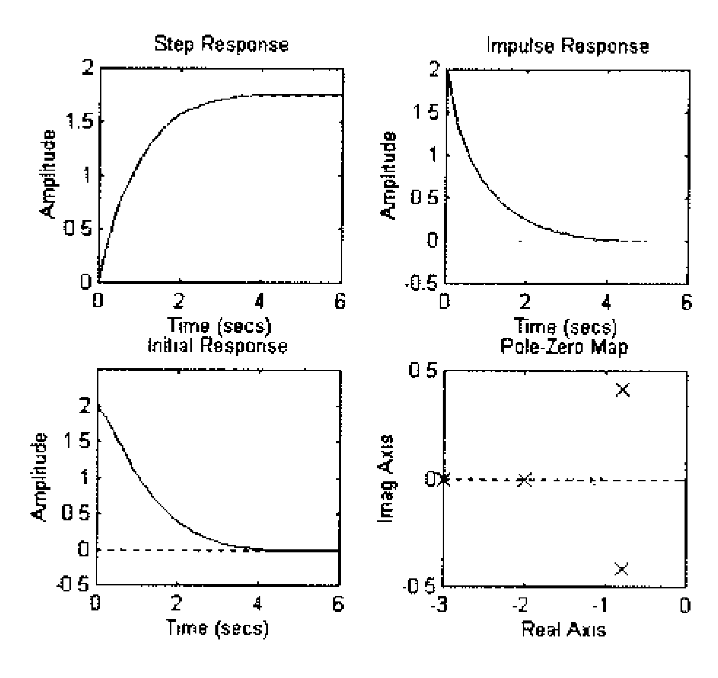


图 3.6 系统的响应与零极点图

例 3.13 多输入多输出系统

$$\begin{cases}\dot{x}=\begin{bmatrix}2.25 & -5 & -1.25 & -0.5\\ 2.25 & -4.25 & -1.25 & -0.25\\ 0.25 & -0.5 & -1.25 & -1\\ 1.25 & -1.75 & -0.25 & -0.75\end{bmatrix}x+\begin{bmatrix}4 & 6\\ 2 & 4\\ 2 & 2\\ 0 & 2\end{bmatrix}u\\ y=\begin{bmatrix}0 & 0 & 0 & 1\\ 0 & 2 & 0 & 2\end{bmatrix}x\end{cases}$$

求单位阶跃响应和单位冲激响应。

解 这是 2 输入 2 输出系统，因此，其阶跃响应和冲激响应各有 4 个。MATLAB 中的 step 和 impulse 函数本身可处理多输入多输出情况，因此，编写 MATLAB 程序是很简单的。

MATLAB 程序为 ex3013.m：

```
% Example 3.13
%
a=[2.25, -5,    -1.25, -0.5; 2.25, -4.25, -1.25, -0.25;
   0.25, -0.5, -1.25, -1;    1.25, -1.75, -0.25, -0.75];
b=[4,6; 2,4; 2,2; 0,2];
c=[0,0,0,1; 0,2,0,2];
d=zeros(2,2);
figure(1)
step(a,b,c,d)
figure(2)
impulse(a,b,c,d)
```

执行后可得如图 3.7 和 3.8 所示的结果曲线。

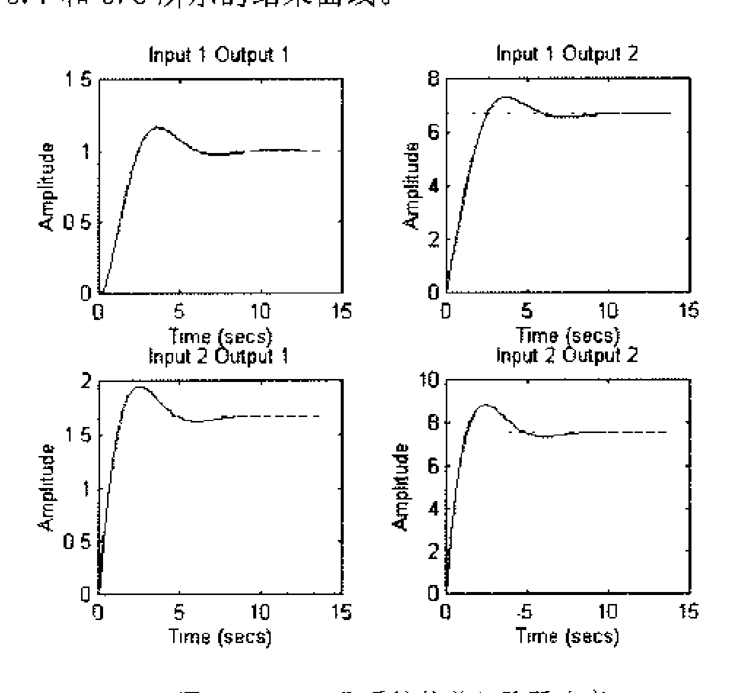


图 3.7 MIMO 系统的单位阶跃响应

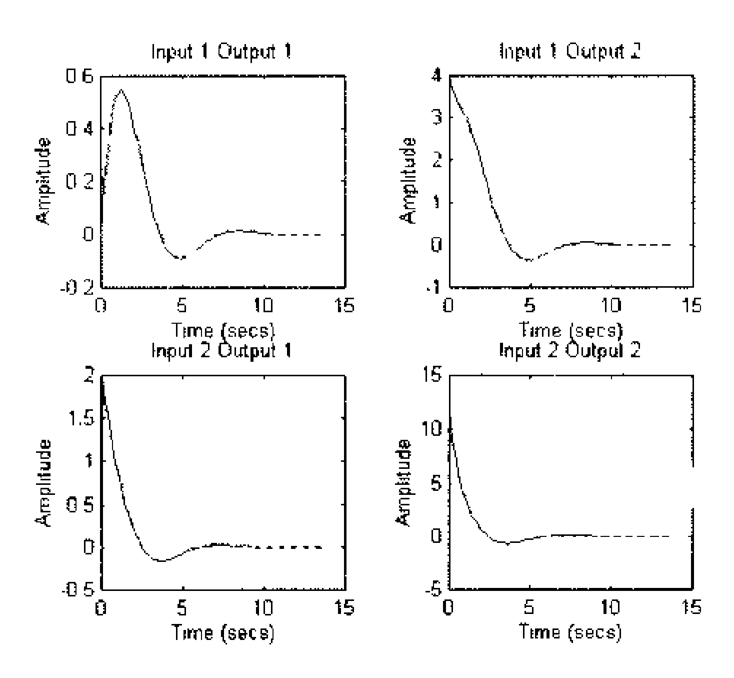


图 3.8 MIMO 系统的单位冲激响应

例 3.14 将例 3.12 中的连续系统，以 t=0.5 取样周期，采用双线性变换算法转换成离散系统，然后求出离散系统的单位阶跃响应、单位冲激响应及零输入响应(设初始状态 $x_0=[1\ 1\ 1\ -1]^T$)。

解 MATLAB 程序为 ex3014.m：

```
% Example 3.14
%
a1=[-1.6,-0.9,0,0; 0.9,0,0,0
     0.4,0.5,-5.0,-2.45; 0,0,2.45,0];
b1=[1;0;1;0];
c1=[1,1,1,1];
d1=[0];
t=0.5;
[a,b,c,d]=c2dm(a1,b1,c1,d1,t, 'tustin');
figure(1)
subplot(2,2,1)
dstep(a,b,c,d)
title( 'Discrete Step Response')
subplot(2,2,2)
dimpulse(a,b,c,d)
title('Discrete Impulse Response')
subplot(2,2,3)
x0=[1;1;1;-1];
dinitial(a,b,c,d,x0)
axis([0 6   0.5 2.5])
```

```
title('Discrete Initial Response')
subplot(2,2,4)
[z,p,k]=ss2zp(a,b,c,d,1);
zplane(z,p)
title('Discrete Pole-Zero Map')
```

执行后得如图 3.9 所示的曲线，它类似于图 3.6。

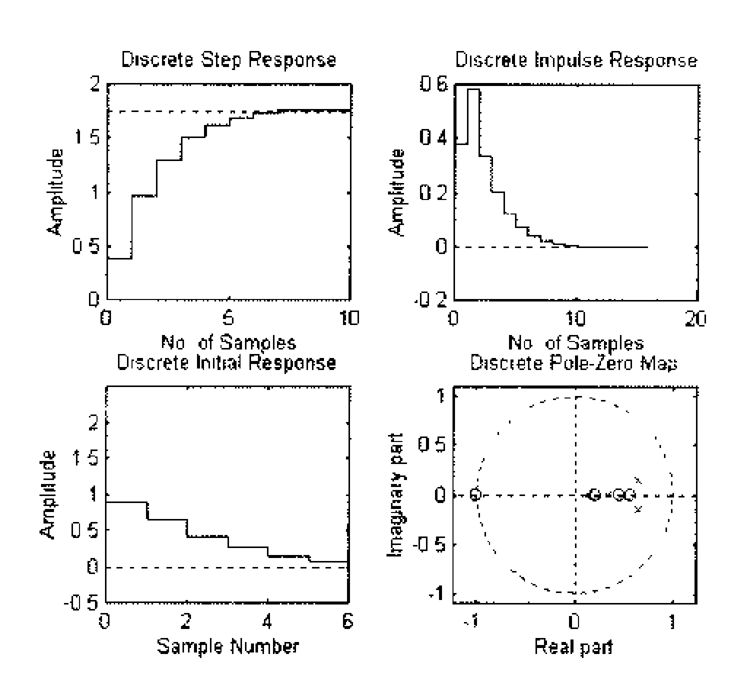


图 3.9 离散系统的响应与零极点图

例 3.15 离散二阶系统

$$H(z)=\frac{0.632}{z^2-1.368z+0.568}$$

求当输入为幅值±1 的方波信号时系统的输出响应。

解 MATLAB 控制系统工具箱提供的 dlsim 函数可用于任意输入下离散系统的仿真，因此 MATLAB 程序为 ex3015.m：

```
% Example 3.15
%
num=0.632;
den=[1,-1.368,.568];
u1=[ones(1,50),-1*ones(1,50)];
u=[u1,u1,u1];
figure(1)
dlsim(num,den,u)
title('Discrete System Simulation')
```

执行后可得如图 3.10 所示的仿真结果。

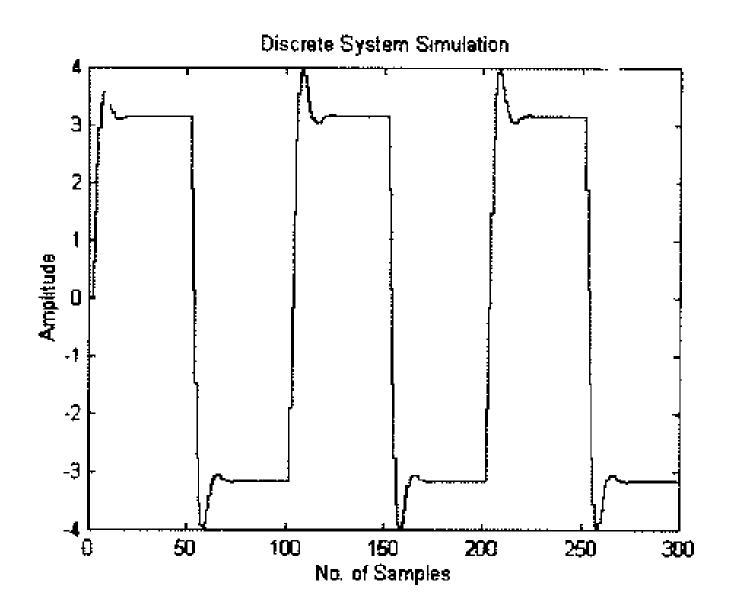


图 3.10　方波输入时离散系统仿真

3.3　控制系统的根轨迹

例 3.16　设开环系统

$$H(s)=\frac{k(3s+1)}{s(2s+1)}$$

绘制出通过单位负反馈构成的闭环系统的根轨迹。

解　可直接利用 rlocus 函数绘制出根轨迹。MATLAB 程序为 ex3016. m：

```
% Example 3.16
%
num=[3 1];
den=[2 1 0];
rlocus(num,den)
title('Root Locus')
```

执行后得如图 3.11 所示的根轨迹。

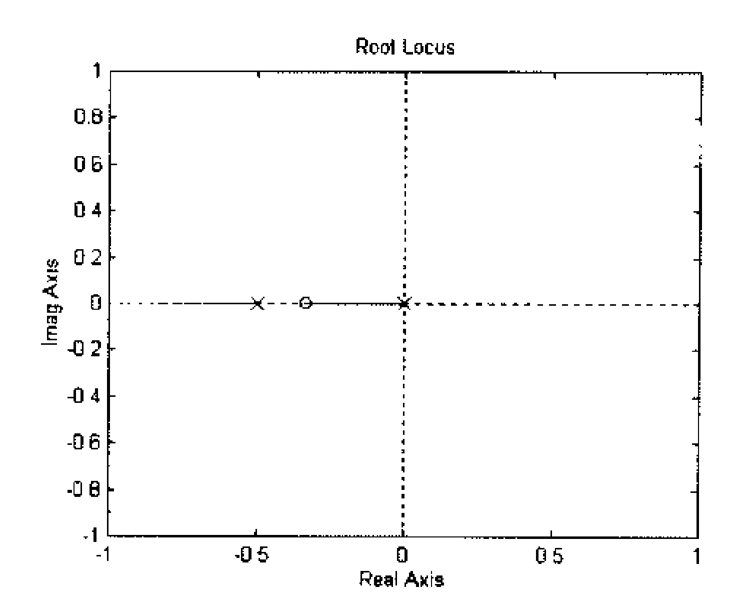
例 3.17　设开环系统

$$H(s)=\frac{k(s+5)}{s(s+2)(s+3)}$$

绘制出闭环系统的根轨迹，并确定交点处的增益 k。

解　利用 rlocus 函数可绘制出根轨迹，而利用 rlocfind 函数可找出根轨迹上任一点处的增益。因此 MATLAB 程序为 ex3017. m：

```
% Example 3.17
%
```



3.11 闭环系统的根轨迹

```
num=[1 5];
den=[1 5 6 0];
rlocus(num,den)
title('Root Locus')
[k,p]=rlocfind(num,den)
gtext('k=0.5')
```

执行时先画出了根轨迹，并提示用户在图形窗口中选择根轨迹上的一点，以计算出增益k及相应的极点。这时将十字光标放在根轨迹的交点处，可得到

```
k=
   0.5072
p=
   -3.2271
   -0.8921
   -0.8808
```

这说明系统有三个极点。实际上，如果能将十字光标准确放在交点时，则p(2)=p(3)。最后得到了如图3.12所示的根轨迹。

例3.18 已知开环传递函数为

$$H(s)=\frac{k}{s^4+16s^3+36s^2+80s}$$

绘出闭环系统的根轨迹。

解 MATLAB程序为ex3018.m：

```
% Example 3.18
%
num=[1];
den=[1 16 36 80 0];
```

```
rlocus(num,den)
title('Root Locus')
```

执行后得如图 3.13 所示的根轨迹。

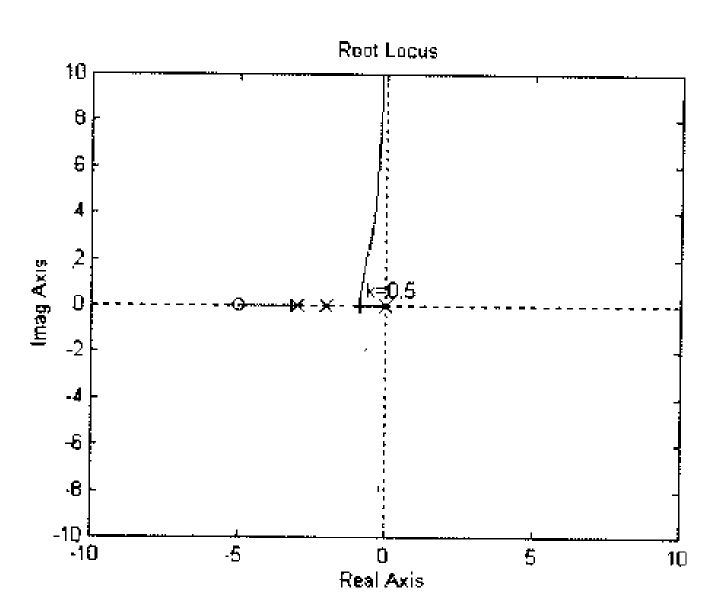


图 3.12 系统的根轨迹

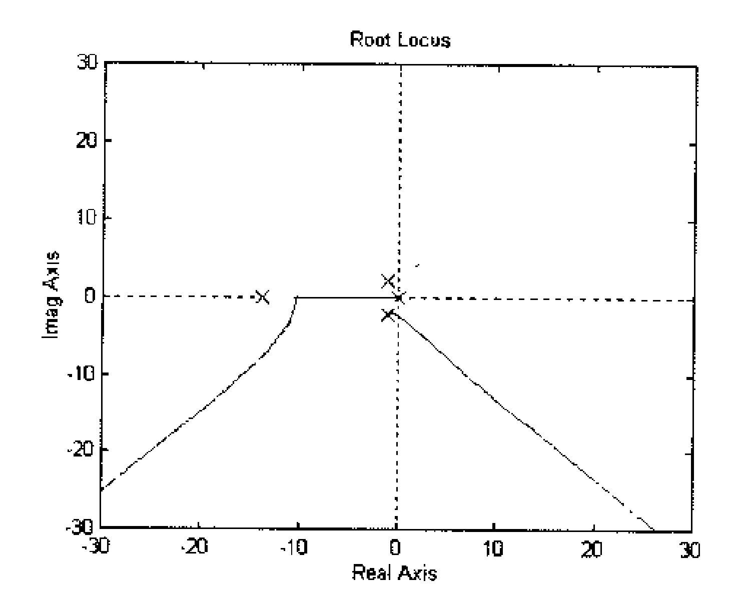


图 3.13 四阶系统根轨迹

例 3.19 已知开环系统传递函数

$$H(s)=\frac{k(s+2)}{(s^2+4s+3)^2}$$

绘制出闭环系统的根轨迹，并分析其稳定性。

解 MATLAB 程序为 ex3019.m：

```
% Example 3.19
%
num=[1 2];
```

```
den1=[1 4 3];
den=conv(den1,den1);
figure(1)
rlocus(num,den)
title('Root Locus')
[k,p]=rlocfind(num,den)

% Checking the stability
figure(2)
k=55;
num1=k*[1 2];
den=[1 4 3];
den1=conv(den,den);
[num,den]=cloop(num1,den1,-1);
impulse(num,den)
title('Impulse Response (k=55)')

% Checking the stability
figure(3)
k=56;
num1=k*[1 2];
den=[1 4 3];
den1=conv(den,den);
[num,den]=cloop(num1,den1,-1);
impulse(num,den)
title('Impulse Response (k=56)')
```

执行后得

```
Select a point in the graphics window

selected_point =
                    0.0000 - 3.1579i

k=
   55.6908
p=
   -5.9878
     0.0029 + 3.1564i
     0.0029 - 3.1564i
```

−2.0179

同时还得到如图 3.14 所示的根轨迹。利用 rlocfind 函数找出根轨迹与虚轴的交点处的增益 k = 55.6，这说明当 k<55.6 时，系统稳定，当 k>55.6 时，系统不稳定。这可借助于 impulse(冲激响应)来说明，我们分别取 k=55 和 k=56 时，求出闭环系统的冲激响应，如图 3.15 和图 3.16 所示，从图中可以清楚看出，当 k=55 时，闭环系统稳定，当 k=56 时，闭环系统发散。

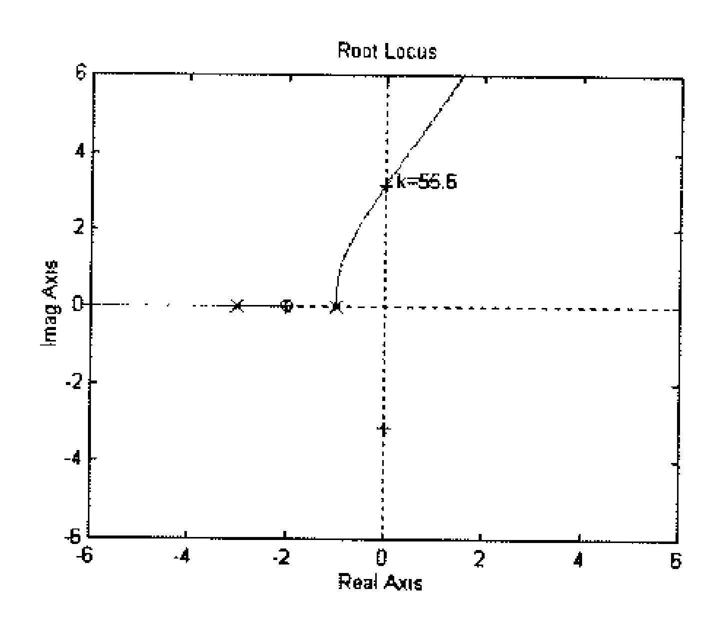


图 3.14　闭环系统根轨迹

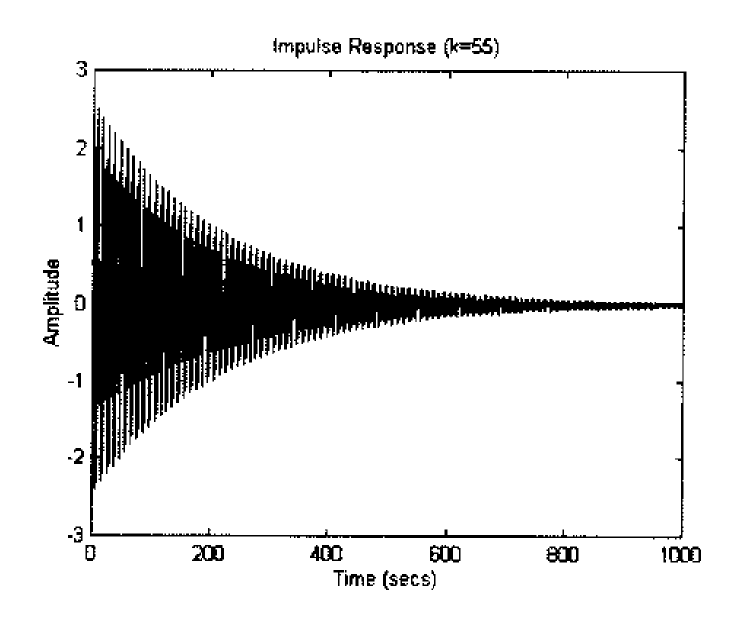


图 3.15　k=55 时闭环系统冲激响应

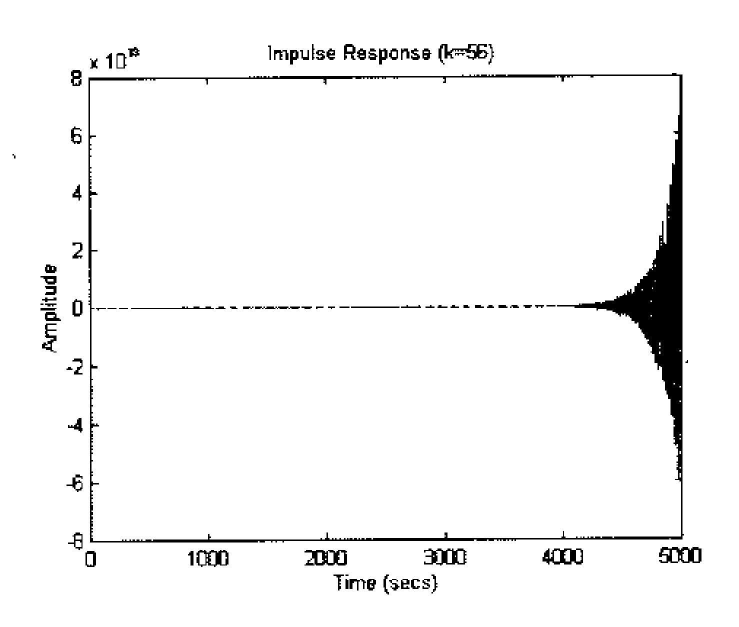


图 3.16　k=56 时闭环系统冲激响应

3.4　控制系统的频域分析

例 3.20　典型二阶系统

$$H(s)=\frac{\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2}$$

绘制出 ξ 取不同值时的 Bode 图。

解　取 $\omega_n=6$，ξ 取[0.1:0.1:1.0]时二阶系统的 Bode 图可直接采用 Bode 得到。MATLAB程序为 ex3020.m：

```
% Example 3.20
%
wn=6;
kosi=[0.1:0.1:1.0];
w=logspace(-1,1,100);
figure(1)
num=[wn.^2];
for kos=kosi
  den=[1 2*kos*wn wn.^2];
  [mag,pha,w1]=bode(num,den,w);
  subplot(2,1,1);hold on
  semilogx(w1,mag);
  subplot(2,1,2);hold on
  semilogx(w1,pha);
```

```
end
subplot(2,1,1);grid on
title('Bode Plot');
xlabel('Frequency (rad/sec)');
ylabel('Gain dB');
subplot(2,1,2);grid on
xlabel('Frequency (rad/sec)');
ylabel('Pbase deg');
hold off
```

执行后得如图 3.17 所示的 Bode 图。

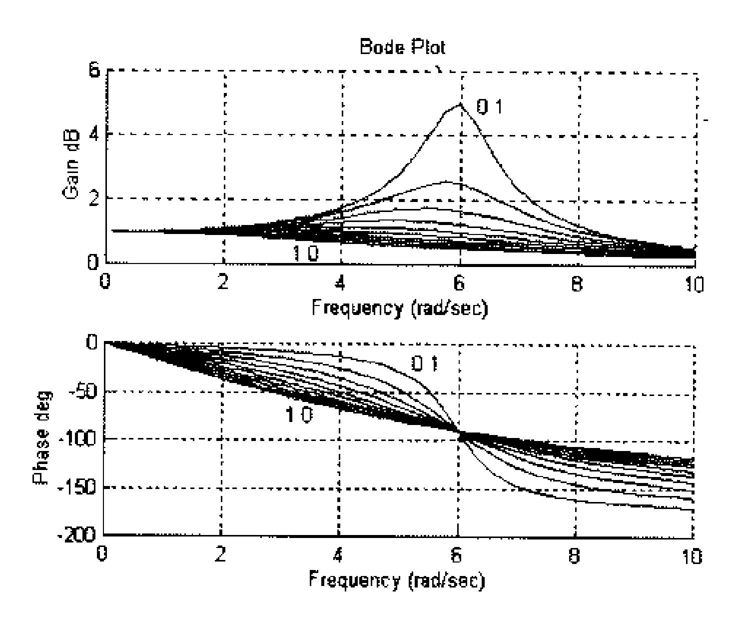


图 3.17　典型二阶系统的 Bode 图

从图 3.17 中可以看出，当 $\omega \to 0$ 时，相角 $\theta(\omega)$ 也趋于 0；当 $\omega \to \infty$ 时，$\theta(\omega) \to -180°$；当 $\omega = \omega_n$ 时，$\theta(\omega) = -90°$。当 $\omega = \omega_n$ 时，频率响应的幅度最大。

例 3.21　有系统

$$H(s) = \frac{100(s+4)}{s(s+0.5)(s+50)^2}$$

绘制出系统的 Bode 图。

解　MATLAB 程序为 ex3021.m：

```
% Example 3.21
%
k=100;
z=[-4];
p=[0 -0.5 -50 -50];
[num,den]=zp2tf(z,p,k);
bode(num,den);
title('Bode Plot');
```

执行后得如图 3.18 所示的 Bode 图。

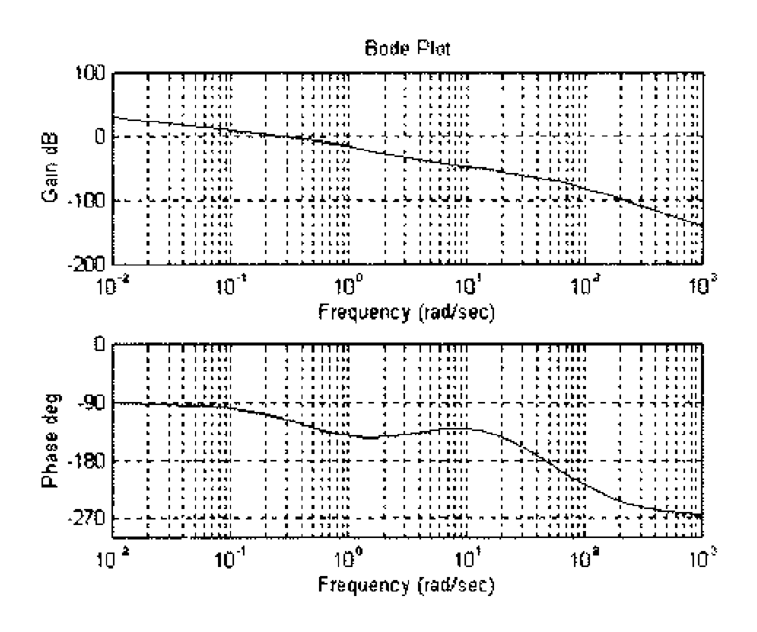


图 3.18 系统 Bode 图

例 3.22 系统传递函数模型为

$$H(s) = \frac{s+1}{(s+2)^3}e^{-0.5s}$$

求出有理传递函数的频率响应，然后在同一张图上绘出以四阶 Pade 近似表示的系统频率响应。

解 Pade 近似可表示 $e^{-0.5s}$，这可由 Pade 函数得到。$(s+2)^3$ 可由 conv(卷积)函数求出。因此，MATLAB 程序为 ex3022.m：

```
% Example 3.22
%
num=[1 1];
den=conv([1 2],conv([1 2],[1 2]));
w=logspace(-1,2);
t=0.5;
[m1,p1]=bode(num,den,w);
p1=p1-t * w' * 180/pi;
[n2,d2]=pade(t,4);
numt=conv(n2,num); dent=conv(den,d2);
[m2,p2]=bode(numt,dent,w);
subplot(2,1,1);
semilogx(w,20 * log10(m1),w,20 * log10(m2),'y - -');
subplot(2,1,2);
semilogx(w,p1,w,p2,'y - -');
subplot(2,1,1);grid on
title('Bode Plot');
```

```
xlabel('Frequency (rad/sec)');
ylabel('Gain dB');
subplot(2,1,2);grid on
xlabel('Frequency (rad/sec)');
ylabel('Phase deg');
```

执行后得如图 3.19 所示的 Bode 图，其中实线为有理传递函数系统的 Bode 图，虚线为带延迟因子系统的 Bode 图。

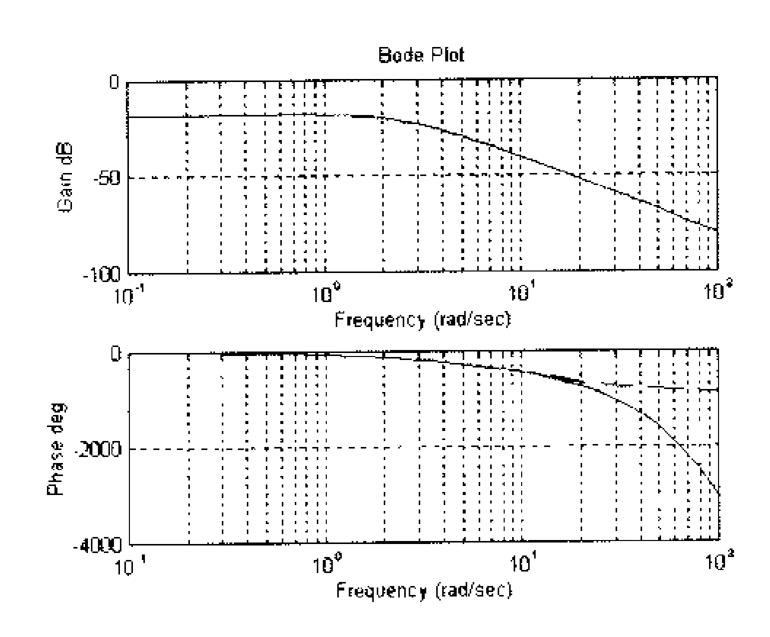


图 3.19 时间延迟系统的 Bode 图

例 3.23 离散二阶系统

$$H(z)=\frac{0.632}{z^2-1.368z+0.568}$$

求系统的离散 Bode 图。

解 MATLAB 程序为 ex3023.m：

```
% Example 3.23
%
num=0.632;
den=[1,-1.368,.568];
dbode(num,den,0.1)
title('Discrete Bode Plot');
```

执行后得如图 3.20 所示的 Bode 图。

例 3.24 开环系统

$$H(s)=\frac{50}{(s+5)(s-2)}$$

绘制出系统的 Nyquist 曲线，并判别闭环系统的稳定性，最后求出闭环系统的单位冲激响应。

解 根据开环系统传递函数，利用 nyquist 函数可绘出系统的 Nyquist 曲线，并根据奈

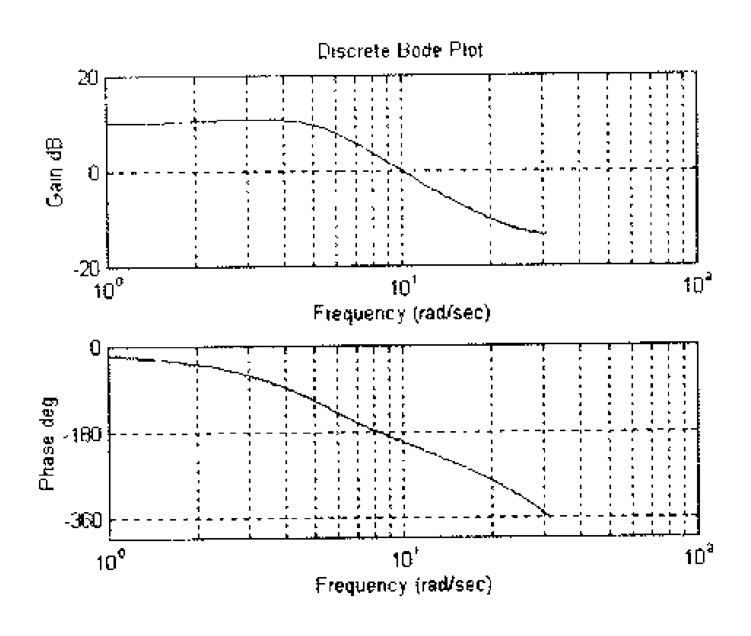


图 3.20 离散系统的 Bode 图

氏判据判别闭环系统的稳定性，最后利用 cloop 函数构成闭环系统，并用 impulse 函数求出冲激响应以验证系统的稳定性结论。

MATLAB 程序为 ex3024.m：

```
% Example 3.24
%
k=50;
z=[];
p=[-5 2];
[num,den]=zp2tf(z,p,k);
figure(1)
nyquist(num,den)
title('Nyquist Plot');
figure(2)
[num1,den1]=cloop(num,den);
impulse(num1,den1)
title('Impulse Response')
```

执行后得如图 3.21 所示的 Nyquist 曲线和如图 3.22 所示的闭环系统单位冲激响应。

从图 3.21 中可以看出，Nyquist 曲线按逆时针包围(−1,0j)点 1 圈，而开环系统包含右半 s-平面上的 1 个极点，因此，以此构成的闭环系统稳定，这可从图 3.22 中得到证实。

例 3.25 开环系统

$$H(s)=\frac{50}{(s+1)(s+5)(s-2)}$$

绘制系统 Nyquist 曲线，判断闭环系统稳定性，绘制出闭环系统的单位冲激响应。

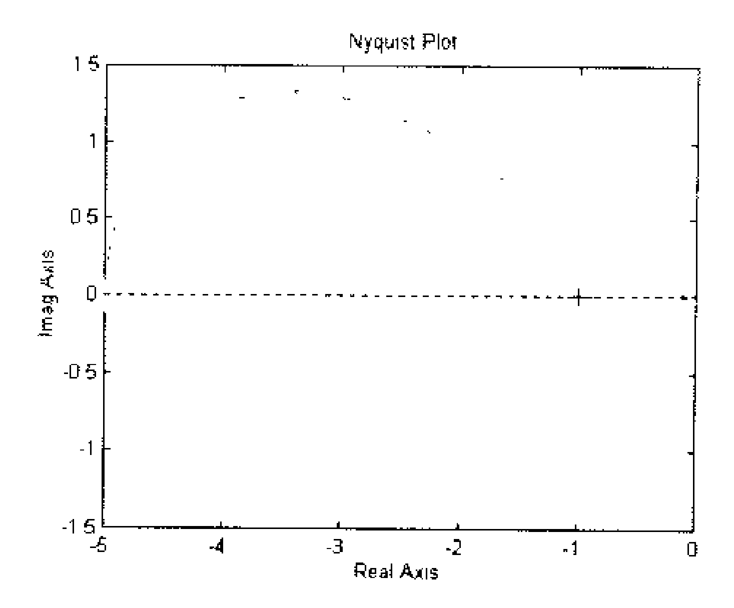


图 3.21　系统 Nyquist 曲线

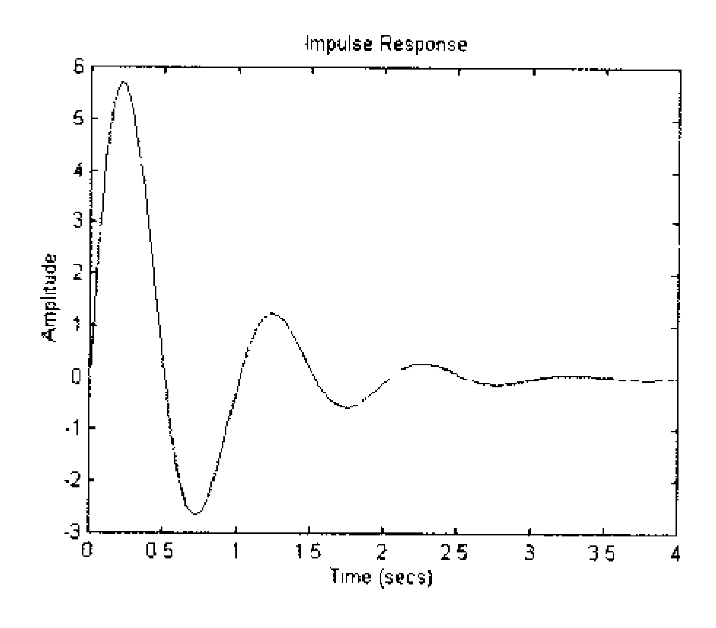


图 3.22　闭环系统单位冲激响应

解　MATLAB 程序为 ex3025.m：

```
% Example 3.25
%
k=50;
z=[];
p=[-1 -5 2];
[num,den]=zp2tf(z,p,k);
figure(1)
nyquist(num,den)
title('Nyquist Plot');
```

```
figure(2)
[num1,den1]=cloop(num,den);
impulse(num1,den1)
title('Impulse Response')
```

执行后得如图 3.23 所示的 Nyquist 曲线和如图 3.24 所示的闭环系统单位冲激响应。

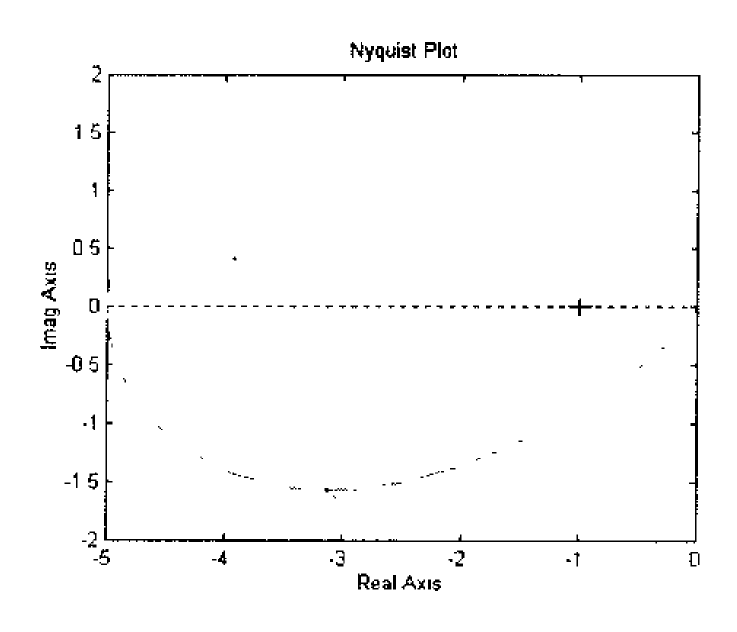


图 3.23　系统 Nyquist 曲线

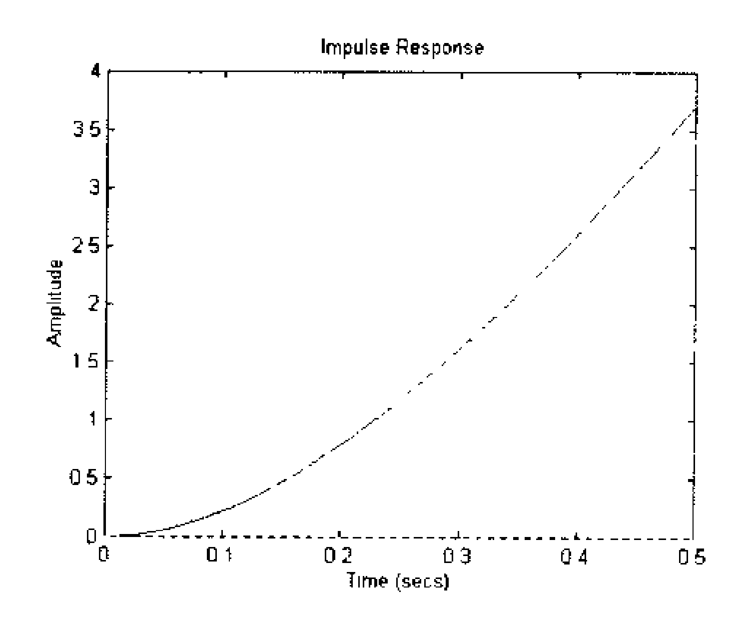


图 3.24　闭环系统单位冲激响应

从图 3.23 中可以看出，系统 Nyquist 曲线按逆时针方向包围(−1,0j)点−1 圈，即按顺时针方向包围(−1,0j)点 1 圈，而开环系统包含右半 s-平面 1 个极点，因此闭环系统不稳定，这可由图 3.24 中得到证实。

例 3.26 线性系统状态空间模型

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ -62.5 & -213.8 & -204.2 & -54 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} u(t) \\ y(t) = [1562 \quad 1875 \quad 0 \quad 0]\, x(t) \end{cases}$$

绘制出系统的 Bode 图和 Nyquist 曲线，判别系统的稳定性，并绘制闭环系统的单位冲激响应进行验证。

解 MATLAB 程序为 ex3026.m：

```
% Example 3.26
%
a=[0 1 0 0; 0 0 1 0; 0 0 0 1; -62.5 -213.8 -204.2 -54];
b=[0 0 0 1]'; c=[1562 1875 0 0]; d=0;
w=logspace(-1,1);
figure(1)
bode(a,b,c,d,1,w)
title('Bode Plot')
figure(2)
nyquist(a,b,c,d,1,w)
title('Nyquist Plot')
[z,p,k]=ss2zp(a,b,c,d);
p
figure(3)
[a1,b1,c1,d1]=cloop(a,b,c,d);
impulse(a1,b1,c1,d1)
title('Impulse Response')
```

执行后得极点位置

```
p=
   -50.0011
   -2.4969
   -1.0027
   -0.4992
```

同时产生如图 3.25～图 3.27 所示的结果曲线。

从图 3.26 可以看出，Nyquist 曲线没有包围点(−1,0j)，而开环系统四个极点均位于左半 s-平面，因此闭环系统稳定，这可由图 3.27 得到证实。

例 3.27 离散系统

$$H(z) = \frac{0.632}{z^2 - 1.368z + 0.568}$$

绘制出系统的 Nyquist 曲线，判别闭环系统稳定性，并绘制出闭环系统的单位冲激响应。

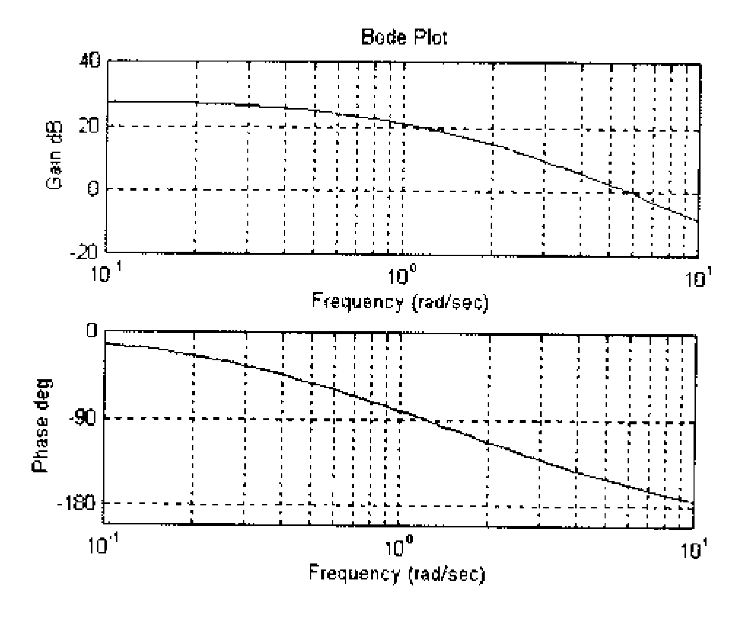


图 3.25 开环系统的 Bode 图

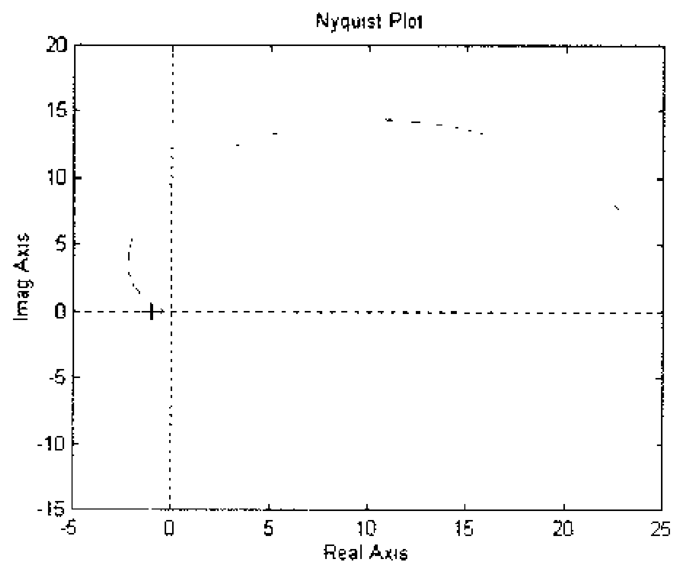


图 3.26 系统的 Nyquist 曲线

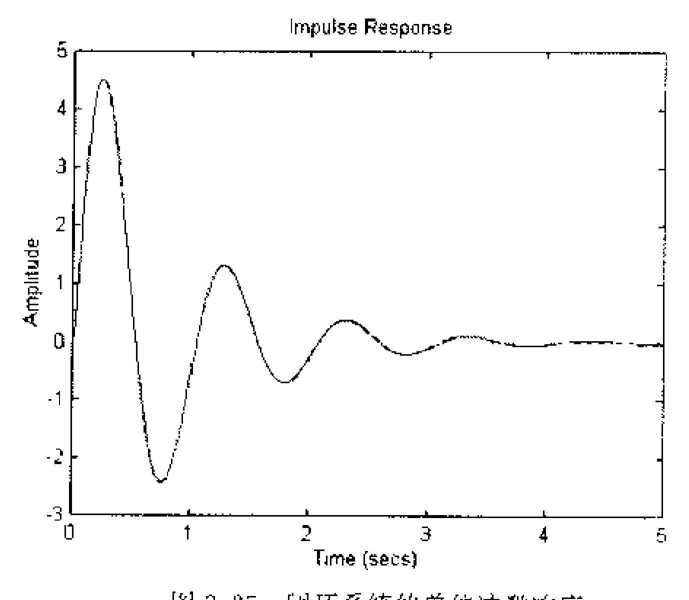


图 3.27 闭环系统的单位冲激响应

解 MATLAB 程序为 ex3027.m：

```
% Example 3.27
%
num=0.632;
den=[1,-1.368,0.568];
[z,p,k]=tf2zp(num,den);
p
dnyquist(num,den,0.1)
title('Discrete Nyquist Plot');
figure(2)
[num1,den1]=cloop(num,den);
dimpulse(num1,den1)
title('Discrete Impulse Response')
```

执行后得开环系统极点位置

```
p=
    0.6840+0.3165i
    0.6840-0.3165i
```

并产生如图 3.28 和 3.29 所示的结果。

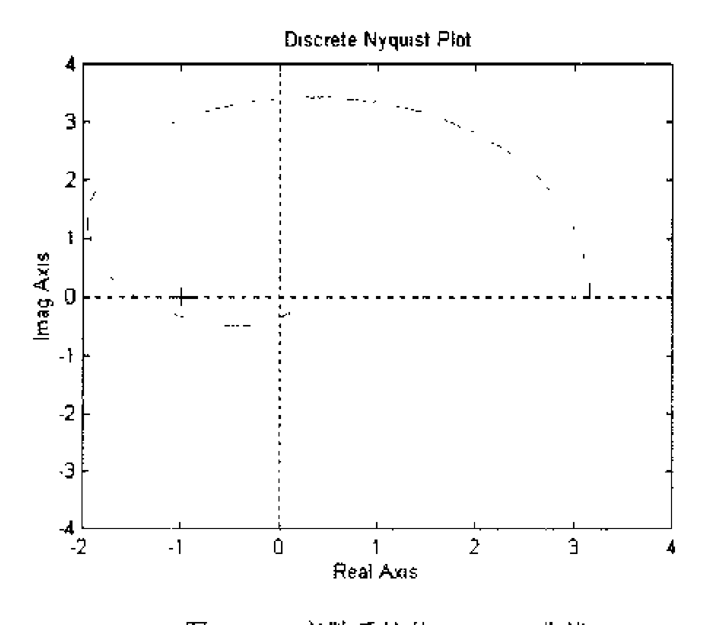


图 3.28 离散系统的 Nyquist 曲线

从图 3.28 可以看出，Nyquist 曲线按逆时针包围(−1,0j)点−2 圈，而开环系统的两个极点均位于单位圆内，因此，闭环系统不稳定，这可从图 3.29 中得到证实。

例 3.28 一多环系统

$$G(s)=\frac{16.7s}{(0.85s+1)(0.25s+1)(0.0625s+1)}$$

其结构如图 3.30 所示，试用 Nyquist 频率曲线判断系统的稳定性。

解 虚线框为一内环，先算出内环传递函数 H(s)

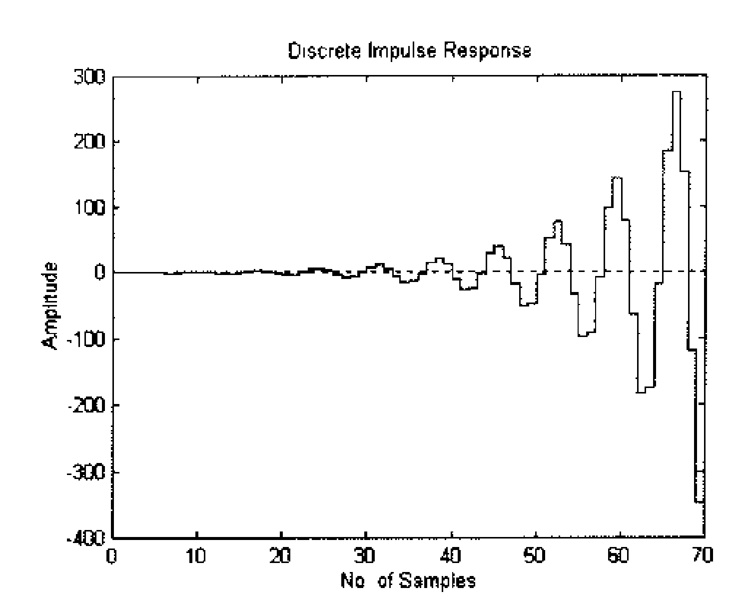


图 3.29　离散闭环系统的单位冲激响应

$$H(s) = \frac{G(s)}{1 + G(s)}$$

然后以 H(s) 为开环传递函数绘制出 Nyquist 曲线，据此可判定闭环系统的稳定性。但这里不能直接采用奈氏判据，因为在前向通道上有一放大系数 k=10，因此奈氏判据中的临界点应改成(−1/k,0j)。

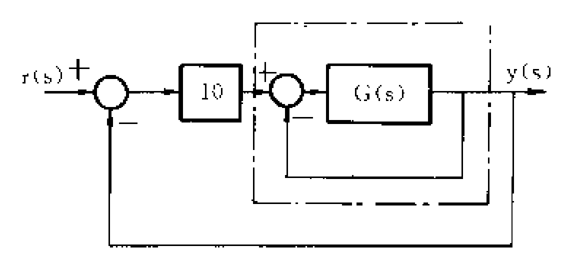


图 3.30　多环系统结构图

MATLAB 程序为 ex3028.m：

```
% Example 3.28
%
k1=16.7/.0125;
z1=[0];
p1=[-1.25 -4 -16];
[num1,den1]=zp2tf(z1,p1,k1);
[num,den]=cloop(num1,den1);
[z,p,k]=tf2zp(num,den);
p
figure(1)
nyquist(num,den)
title('Nyquist Plot')
figure(2)
[num2,den2]=cloop(num,den);
impulse(num2,den2)
title('Impulse Response')
```

执行后得极点位置

p=

−10.5969−36.2148i

−10.5969−36.2148i

−0.0562

并得到如图 3.31 和 3.32 所示的结果曲线。

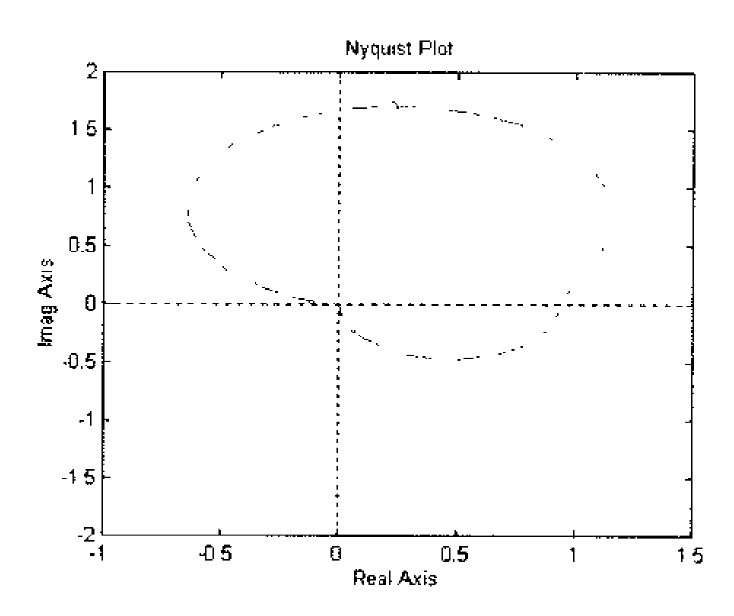


图 3.31　多环系统的 Nyquist 曲线

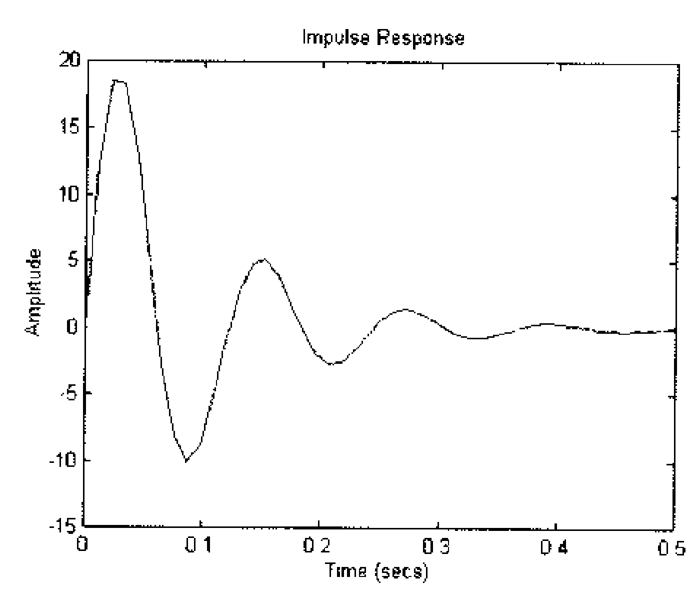


图 3.32　多环系统的闭环单位冲激响应

从图 3.31 中可以看出，Nyquist 曲线不包围(−0.1,0j)点，而开环系统的三个极点均位于左半 s-平面，因此这个系统是稳定的，这可从图 3.32 中得到证实。

例 3.29　非线性系统

$$\begin{cases} \dot{x} = ax + bu \\ u = f(\sigma) \\ \sigma = cx \end{cases}$$

其中

$$a=\begin{bmatrix}-2.1 & -1.87 & -0.203 & 0.894\\ 1.0 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\qquad b=\begin{bmatrix}1\\0\\0\\0\end{bmatrix}\qquad c=[0\quad 0\quad -1\quad -2]$$

$$k_1\sigma\leqslant f(\sigma)\leqslant k_2\sigma$$

当$[k_1,k_2]$的(1/2,2/3)或[1,2]时，判断非线性系统的绝对稳定性。

解　这是典型的鲁里叶问题，其绝对稳定性可由修改后的奈氏判据判定。

非线性系统的线性部分传递函数可由拉氏变换求得

$$\sigma=-w(s)u$$

$$w(s)=-c(sI-a)^{-1}b$$

这可由 ss2tf 函数得到，注意这里求得的结果为$H(s)=c(sI-a)^{-1}b+d$，因此可令 $d=0$，并将结果取负就可得到 $w(s)$。

由这一传递函数构成了如图 3.33 所示的等效结构图。

然后以 $w(s)$为开环传递函数，绘制为 Nyquist 曲线，最后根据修改后的奈氏判据判定系统的绝对稳定性。

奈氏判据：如果开环传递函数 $w(s)$有 v 个极点位于右半 s -平面，则当 Nyquist 曲线按逆时针方向包围线段$(-1/k_1,-1/k_2)$ v 圈，且不与线段相交，则非线性系统绝对稳定。

图 3.33　非线性系统等效结构

MATLAB 程序为 ex3029.m：

```
% Example 3.29
%
a=[-2.1 -1.87 -0.203 0.894 ; 1 0 0 0; 0 1 0 0; 0 0 1 0];
b=[1 0 0 0]';
c=[0 0 -1 -2];
d=0;
[num,den]=ss2tf(a,b,c,d);
num=-1 * num;
[z,p,k]=tf2zp(num,den);
p
figure(1)
nyquist(num,den)
title('Nyquist Plot')
```

执行后得极点位置

```
p=
    -0.7000+1.0000i
    -0.7000-1.0000i
```

```
    -1.2000
     0.5000
```

同时得到如图 3.34 所示的 Nyquist 曲线。

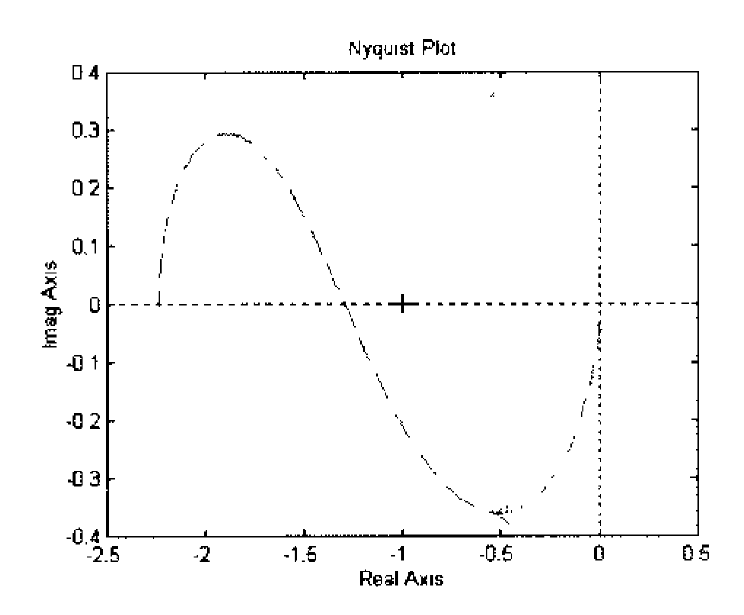


图 3.34　非线性系统的 Nyquist 曲线

从图 3.34 中可以看出，当$[k_1,k_2]$取$[1/2,2/3]$时，Nyquist 曲线按逆时针方向包围线段$(-1/k_1,-1/k_2)$ 1 圈，而开环系统有一极点在位于右半 s-平面，因此这时系统绝对稳定。当$[k_1,k_2]$取$[1,2]$时，Nyquist 曲线按逆时针方向包围线段$[-1/k_1,-1/k_2]$ −1 圈，故系统不稳定。

例 3.30　系统开环传递函数为

$$H(s)=\frac{16.7s}{(0.8s+1)(0.25s+1)(0.0625s+1)}$$

绘制出 Nichols 图。

解　MATLAB 程序为 ex3030.m：

```
% Example 3.30
%
k1=16.7/.0125;
z1=[0];
p1=[-1.25 -4 -16];
[num,den]=zp2tf(z1,p1,k1);
figure(1)
ngrid('new')
nichols(num,den)
title('Nichols Plot')
```

执行后得如图 3.35 所示的 Nichols 图。

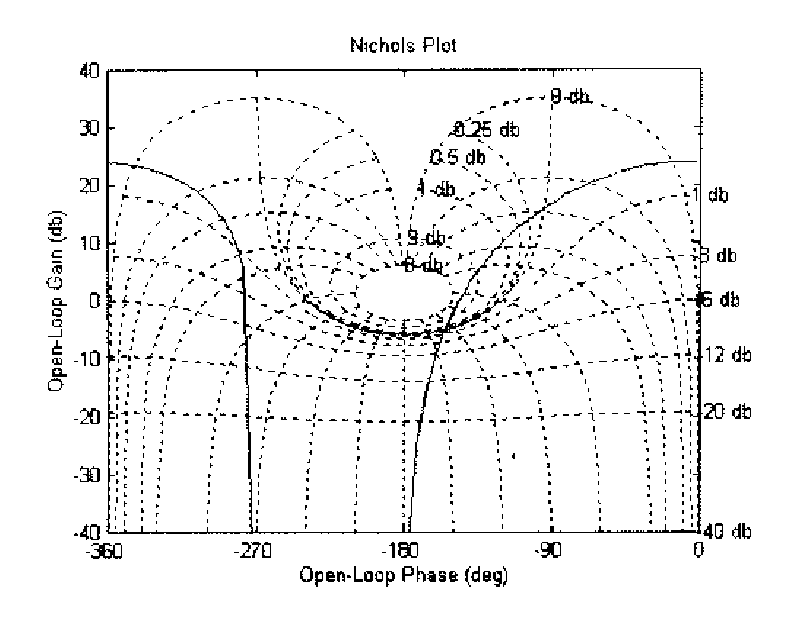


图 3.35　系统的 Nichols 图

3.5　极点配置和观测器设计

例 3.31　被控对象

$$H(s)=\frac{10}{(s+1)(s+2)(s+3)}$$

设计反馈控制器 $u=-kx$，使闭环系统的极点为 $\mu_1=-2+j2\sqrt{3}$，$\mu_2=-2-j2\sqrt{3}$，$\mu_3=-10$。

解　定义状态变量

$$x_1=y$$
$$x_2=\dot{x}_1$$
$$x_3=\dot{x}_2$$

因此，系统的状态方程为

$$\begin{cases}\dot{x}=\begin{bmatrix}0&1&0\\0&0&1\\-6&-11&-6\end{bmatrix}x+\begin{bmatrix}0\\0\\10\end{bmatrix}u\\ y=\begin{bmatrix}1&0&0\end{bmatrix}x\end{cases}$$

然后根据变换法和 Ackermenn 公式求出所设计的增益阵 k，这里先给出按这两种方法求解的 MATLAB 程序，然后给出以工具箱函数 place（或 acker）求解的结果，它们是一致的。

MATLAB 程序为 ex3031.m：

```
% Example 3.31
%
```

```
% Pole placement - - using transformation matrix
%
disp('Pole placement - - using transformation matrix')
a=[0 1 0; 0 0 1; -6 -11 -6];
b=[0; 0; 10];
% Step 1
cam=ctrb(a,b);
disp('The rank of controllability matrix')
rc=rank(cam)
% Step 2
beta=poly(a)
% Step 3
a1=beta(2); a2=beta(3); a3=beta(4);
w=[a2 a1 1; a1 1 0; 1 0 0];
t=cam * w;
% Step 4
j=[-2+2 * sqrt(3) * i   0       0
        0      -2-2 * sqrt(3) * i 0
        0                    0  -10];
alph=poly(j)
aa1= alph(2); aa2=alph(3); aa3=alph(4);
% Step 5
k=[aa3-a3 aa2-a2 aa1-a1] * (inv(t))
%
% Pole placement - - using Ackermann's formula
%
disp('Pole placement - - using Ackermanns formula')
a=[0 1 0; 0 0 1; -6 -11 -6];
b=[0; 0; 10];
% Step 1
cam=ctrb(a,b);
disp('The rank of controllability matrix')
rc=rank(cam)
% Step 2
j=[-2+2 * sqrt(3) * i   0       0
        0      -2-2 * sqrt(3) * i 0
        0                    0  -10];
alph=poly(j);
% Step 3
```

```
phi=polyvalm(alph,a);
% Step 4
k=[0 0 1]*(inv(cam))*phi
%
% Pole placement - - using place function in MATLAB
%
disp('Pole placement - - using place function in MATLAB')
p=[-2+2*sqrt(3)*i -2-2*sqrt(3)*i -10];
k=place(a,b,p)
```

执行后得

```
Pole placement - - using transformation matrix

The rank of controllability matrix
rc =
     3
beta =
        1.0000    6.0000   11.0000    6.0000
alph =
        1.0000   14.0000   56.0000  160.0000
k =
    15.4000    4.5000    0.8000

Pole placement - - using Ackermanns formula

The rank of controllability matrix
rc =
     3
k =
    15.4000    4.5000    0.8000

Pole placement - - using place function in MATLAB

place: ndigits= 17
k =
    15.4000    4.5000    0.8000
```

由此可以看出，可控性矩阵 cam 的秩为 3，因此系统完全可控。这三种方法求出的增益阵 k 是一致的，其最为简洁的方法是利用 MATLAB 工具箱函数进行求解。因此在后续设计示例中，一般都采用 place 或 acker 进行设计。

例 3.32 含积分环节的类型 1 伺服系统设计，设对象为

$$H(s) = \frac{1}{s(s+1)(s-2)}$$

设计控制器 $u=-kx+k_1r$，使闭环系统具有极点 $-2\pm j2\sqrt{3}$，-10。

解 定义状态变量

$$x_1 = y$$
$$x_2 = \dot{x}_1$$
$$x_3 = \dot{x}_2$$

则系统可写成

$$\begin{cases} \dot{x} = ax + bu \\ y = cx \end{cases}$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -2 & -3 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad c = [1 \quad 0 \quad 0]$$

采用如图 3.36 所示的控制结构，即

$$u = -kx + k_1 r \qquad k = [k_2 \quad k_3 \quad k_4]$$

然后采用 place 函数直接设计。

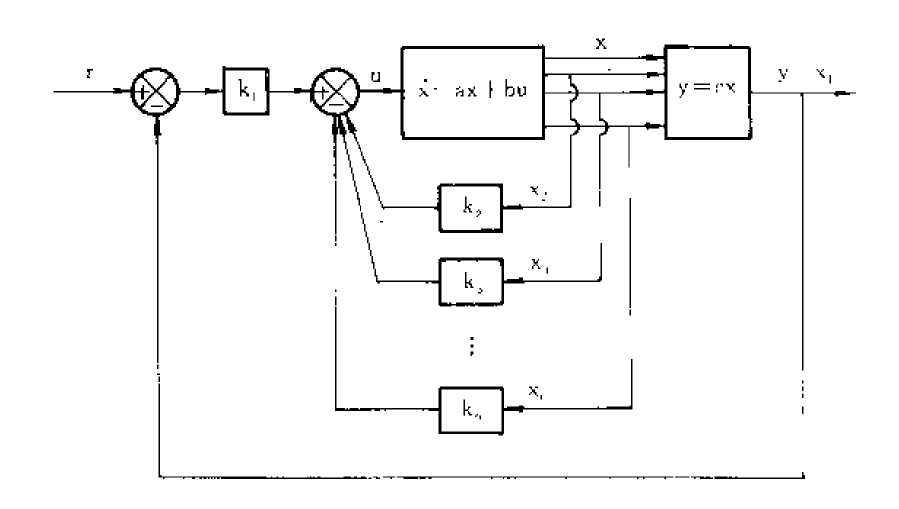


图 3.36 类型 1 伺服系统的一般结构

MATLAB 程序为 ex3032.m：

```
% Example 3.32
%
% Pole placement - - using place function in MATLAB
%
disp('Pole placement -- using place function in MATLAB')
a=[0 1 0; 0 0 1; 0 -2 -3];
b=[0; 0; 1];
c=[1 0 0];
d=[0];
```

```
disp('The rank of controllability matrix')
rc=rank(ctrb(a,b))
p=[-2+2*sqrt(3)*i -2-2*sqrt(3)*i -10];
k=place(a,b,p)
a1=a-b*k;
b1=b*k(1);
c1=c; d1=d;
figure(1)
step(a1,b1,c1,d1)
title('Step Response of Designed Servo System')
figure(2)
[y,x,t]=step(a1,b1,c1,d1);
plot(t,x)
title('Step Response Curves for x1,x2,x3')
grid on
```

执行后可得

```
Pole placement - - using place function in MATLAB
The rank of controllability matrix
rc =
        3
place: ndigits= 17
k =
    160.0000   54.0000   11.0000
```

并得到了如图 3.37 所示的输出单位阶跃响应和如图 3.38 所示的各状态单位阶跃响应。

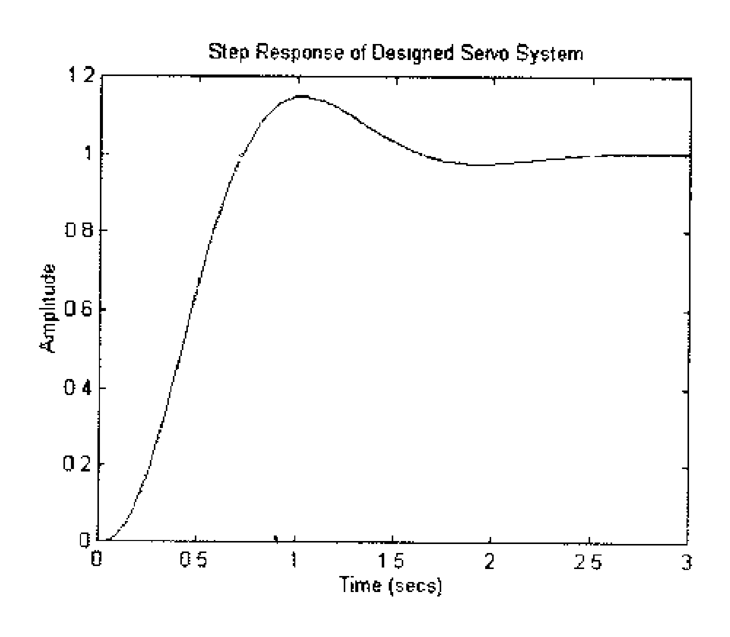


图 3.37　输出单位阶跃响应

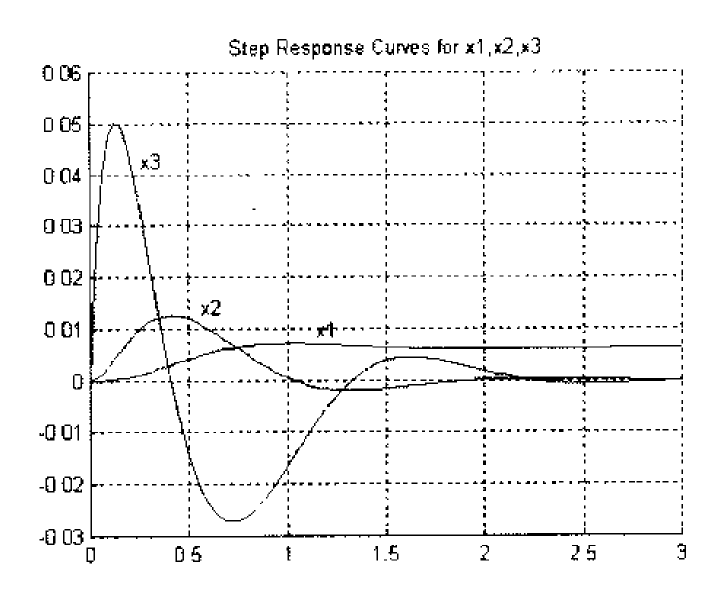


图 3.38　状态单位阶跃响应

例 3.33　不含积分环节的类型 1 伺服系统设计。对车载倒立摆系统实现控制，使小车位置作为输出的闭环系统具有极点：$-1\pm j\sqrt{3}$，-5，-5，-5。

解　先建立倒立摆系统模型。车载倒立摆系统如图 3.39 所示。

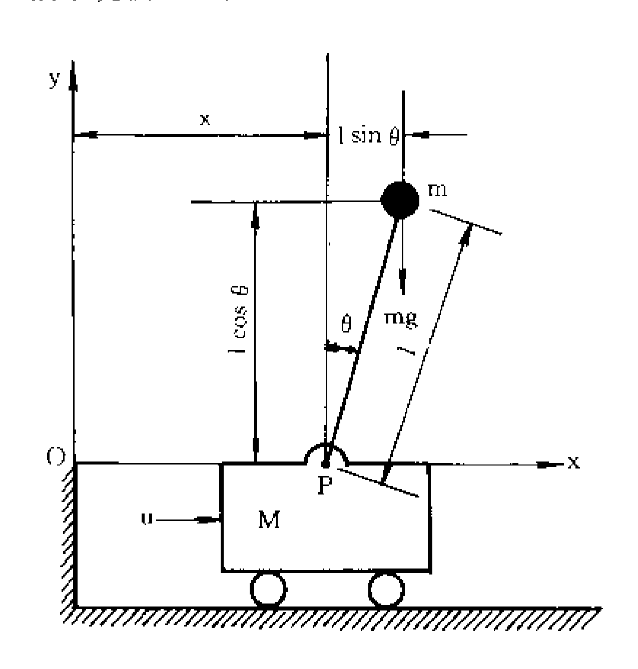


图 3.39　车载倒立摆系统

根据图 3.39 和牛顿第二定律，可得线性化模型

$$\begin{cases} Ml\ddot{\theta} = (M+m)g\theta - u \\ M\ddot{x} = u - mg\theta \end{cases}$$

定义状态变量

$$x_1 = \theta$$
$$x_2 = \dot{\theta}$$
$$x_3 = x$$
$$x_4 = \dot{x}$$

并取 $y=x=x_3$，则有状态方程

$$\begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{M+m}{Ml}g & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{m}{M}g & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 \\ -\frac{1}{Ml} \\ 0 \\ \frac{1}{M} \end{bmatrix} u \\ y = [0 \quad 0 \quad 1 \quad 0] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \end{cases}$$

在本例中取 $M=2$ kg，$m=0.1$ kg，$l=0.5$ m，则状态方程变为

$$\dot{x} = ax + bu$$
$$y = cx + du$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 20.601 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -0.4905 & 0 & 0 & 0 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} \quad c = [0 \quad 0 \quad 1 \quad 0] \quad d = 0$$

注意，这里的 x 为矢量，与上面的位置 x 不同。

倒立摆系统可利用极点配置方法进行设计，实际上这是一种不含积分环节的类型 1 伺服系统，为此在前向通道引入一积分器，对输出 y 也即小车位置值进行积累作用，如图 3.40 所示。

由此得到了

$$\begin{cases} \dot{x} = ax + bu \\ y = cx \\ u = -kx + k_I\xi \\ \dot{\xi} = r - y = r - cx \end{cases}$$

记 $e = \begin{bmatrix} x_e(t) \\ \xi_e(t) \end{bmatrix} = \begin{bmatrix} x(t) - x(\infty) \\ \xi(t) - \xi(\infty) \end{bmatrix}$，$u_e(t) = -kx_e(t) + k_I\xi_e(t)$，由上式可得到

$$\begin{cases} \dot{e} = \hat{a}e + \hat{b}u_e \\ u_e = -\hat{k}e \end{cases}$$

其中

$$\hat{a} = \begin{bmatrix} a & 0 \\ -c & 0 \end{bmatrix} \quad \hat{b} = \begin{bmatrix} b \\ 0 \end{bmatrix} \quad \hat{k} = [k \quad -k_I]$$

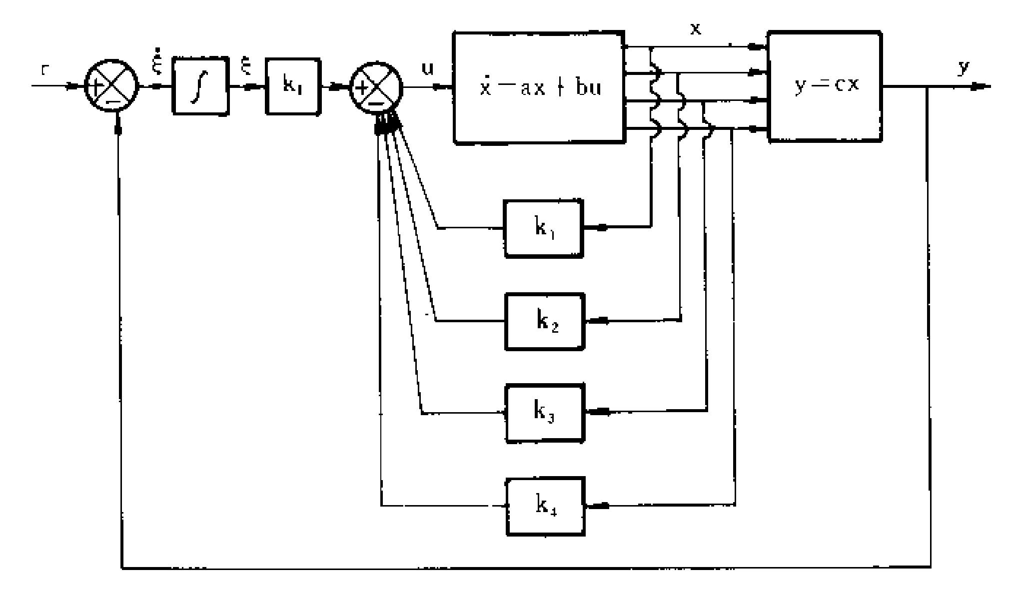


图 3.40 倒立摆系统极点配置结构

本例中

$$\hat{a}=\begin{bmatrix}0 & 1 & 0 & 0 & 0\\ 20.601 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0\\ -0.4905 & 0 & 0 & 0 & 0\\ 0 & 0 & -1 & 0 & 0\end{bmatrix} \qquad \hat{b}=\begin{bmatrix}0\\ -1\\ 0\\ 0.5\\ 0\end{bmatrix}$$

因此设计问题变成了对上述误差状态方程的设计，使闭环系统具有指定的性能。如本例指定的极点$-1\pm j\sqrt{3}$，-5，-5，-5，可使系统具有设置时间4～5秒，最大超调量15%～16%。

因此利用MATLAB的place函数进行设计，然后绘制出系统的单位阶跃响应。MATLAB程序为ex3033.m：

```
% Example 3.33
%
% Design of an inverted pendulum control system
% using pole placement - - Ackermann
%
% Modelling
a=[0 1 0 0; 20.601 0 0 0; 0 0 0 1; -0.4905 0 0 0];
b=[0; -1; 0; 0.5];
c=[0 0 1 0];
d=[0];
a1=[a zeros(4,1); -c 0];
b1=[b;0];
% Check the controllability
disp('The rank of controllability matrix')
```

```
rc=rank(ctrb(a1,b1))
% Design
p=[-1+sqrt(3)*i  -1-sqrt(3)*i  -5  -5  -5];
k=acker(a1,b1,p)
%
% Step response
%
% The close loop state system is denoted as (ac,bc,cc,dc)
%
k1=k(1:4); ki=-k(5);
ac=[a-b*k1 b*ki; -c 0];
bc=[0 0 0 0 1]';
cc=[c 0];
dc=[0];
figure(1)
v=[0 5 -0.4 1.4];
step(ac,bc,cc,dc)
axis(v)
title('Step Response of inverted pendulum system')
xlabel('Sec')
ylabel('Output y=x3')
figure(2)
[y,x,t]=step(ac,bc,cc,dc);
plot(t,x,'y')
axis(v)
title('Step Response Curves for x1,x2,x3,x4,x5')
xlabel('Sec')
ylabel('x1, x2, x3, x4, x5')
text(1.2, 0.05, 'x1')
text(1.2, -0.33, 'x2')
text(1.2, 0.4, 'x3')
text(2.2, 0.25, 'x4')
text(1.5, 1.28, 'x5')
```

执行后得

```
The rank of controllability matrix
rc=
   5
k=
  -157.6336  -35.3733  -56.0652  -36.7466  50.9684
```

因此系统完全可控，并设计得到 $k_f=-k(5)=-50.9684$，$k_1=-157.6336$，$k_2=-35.3733$，$k_3=-56.0652$，$k_4=-36.7466$。同时还得到了如图 3.41 和图 3.42 所示的单位阶跃响应曲线。

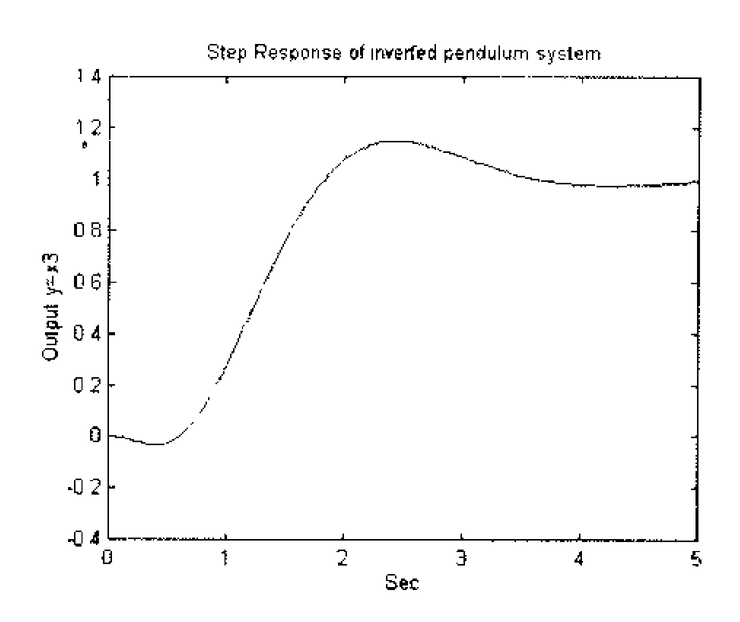


图 3.41　倒立摆系统输出单位阶跃响应

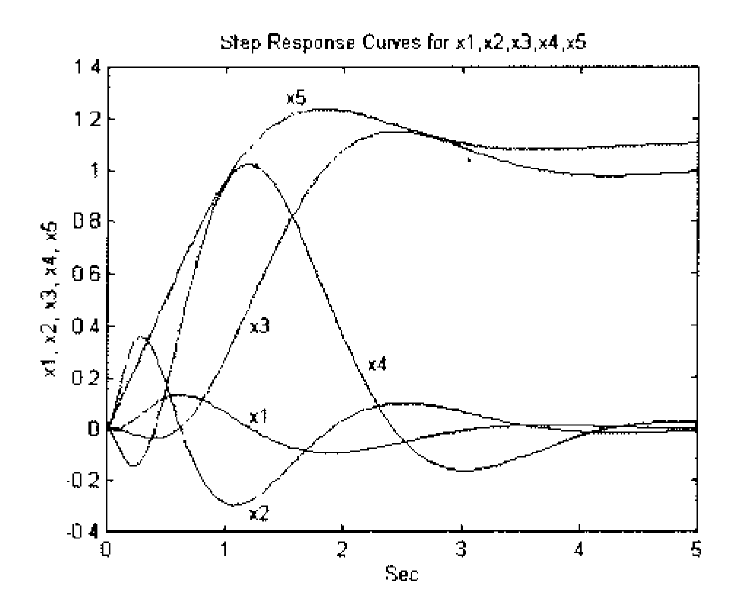


图 3.42　倒立摆系统状态单位阶跃响应

例 3.34　开环系统

$$\begin{cases} \dot{x} = ax + bu \\ y = cx \end{cases}$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -6 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad c = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

设计全阶状态观测器，使观测器的闭环极点为$-2\pm j2\sqrt{3}$，-5。

解 状态观测器的结构见第1章图1.3。为求出状态观测器的反馈增益阵k_e，先为原系统构造一对偶系统

$$\begin{cases}\dot{p}=a'p+c'v\\ q=b'p\end{cases}$$

然后采用极点配置方法对对偶系统进行闭环极点位置的配置，得到反馈增益阵k，从而可由对偶原理得到原系统的状态观测器增益阵$k_e=k'$。

MATLAB程序为ex3034.m：

```
% Example 3.34
%
% Design of a full-order state observer
%
a=[0 1 0; 0 0 1; -6 -11 -6];
b=[0; 0; 1];
c=[1 0 0];
% Check observability
disp('The rank of observability matrix')
ro=rank(obsv(a,c))
% Construct the daul system (a1,b1,c1)
a1=a';
b1=c';
c1=b';
% Solve K using poles placement --Ackermann
p=[-2+2*sqrt(3)*i  -2-2*sqrt(3)*i  -5];
k=acker(a1,b1,p);
% Obtain the state observer gain Ke
ke=k'
```

执行后得

```
The rank of observability matrix
ro=
      3
ke=
      3.0000
      7.0000
     -1.000
```

由于ro=3，所以系统可观，因此可设计全阶状态观测器。

例3.35 被控对象

$$\begin{cases}\dot{x}=\begin{bmatrix}0 & 1\\ 20.6 & 0\end{bmatrix}x+\begin{bmatrix}0\\ 1\end{bmatrix}u\\ y=\begin{bmatrix}1 & 0\end{bmatrix}x\end{cases}$$

设计调节器使闭环极点为－1.8±j2.4，而且状态不可测，因此设计状态观测器使其闭环极点为－8，－8。

解　调节器和观测器的设计分开进行，状态观测器的设计借助于对偶原理。

在设计之前，应判别其可控性和可观性。

MATLAB 程序为 ex3035.m：

```
% Example 3.35
%
% Design of regulator and state observer
%
a=[0 1; 20.6 0];
b=[0; 1]; c=[1 0];

% Check controllability and observability
disp('The rank of controllability matrix')
rc=rank(ctrb(a,b))
disp('The rank of observability matrix')
ro=rank(obsv(a,c))

% Design regulator
p=[-1.8+2.4*i,-1.8-2.4*i];
k=acker(a,b,p)

% Design state observer
a1=a'; b1=c'; c1=b';
p1=[-8 -8];
k1=acker(a1,b1,p1);
ke=k1'
```

执行后得

```
The rank of controllability matrix
rc =
      2

The rank of observability matrix
ro =
      2

k =
    29.6000    3.6000
```

```
ke =
        16.0000
        84.6000
```

例 3.36 最小阶状态观测器设计。设开环系统

$$\begin{cases} \dot{x} = ax + bu \\ y = cx \end{cases}$$

其中

$$a = \begin{bmatrix} 0 & \vdots & 1 & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & \vdots & 0 & 1 \\ -6 & \vdots & -11 & -6 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad c = [1 \quad 0 \quad 0]$$

假定输出 y(即 x_1)可测，设计最小阶状态观测器，使闭环极点为$-2\pm j2\sqrt{3}$。

解 由于 x_1 可测，因此只需设计 x_2 和 x_3 的状态观测器，故原系统可分成两部分：

可测部分

$$\dot{x}_a = a_{aa}x_a + a_{ab}x_b + b_a u$$

不可测部分

$$\dot{x}_b = a_{ba}x_a + a_{bb}x_b + b_b u$$

其中

$$a_{aa} = 0 \qquad a_{ab} = [1 \quad 0]$$

$$a_{ba} = \begin{bmatrix} 0 \\ -6 \end{bmatrix} \qquad a_{bb} = \begin{bmatrix} 0 & 1 \\ -11 & -6 \end{bmatrix}$$

$$b_a = 0 \qquad b_b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

根据第 1 章的结论，等效系统为

$$\begin{cases} \dot{p} = a_c p + b_c v \\ q = c_c p \end{cases}$$

其中

$$a_c = a_{bb} \qquad b_c v = a_{ba}x_a + b_b u \qquad c_c = a_{ab}$$

因此可方便地利用 MATLAB 的 acker 或 place 函数进行设计。

MATLAB 程序 ex3036.m：

```
% Example 3.36
%
% Design of a minimum-order observer
%
a=[0 1 0; 0 0 1; -6 -11 -6];
b=[0; 0; 1]; c=[1 0 0];
aaa=[a(1,1)]; aab=[a(1,2:3)];
aba=[a(2:3,1)]; abb=[a(2:3,2:3)];
ba=b(1,1); bb=b(2:3,1);
```

```
% subsystem
a1=abb; c1=aab;
disp('The rank of observability matrix')
ro=rank(obsv(a1,c1))

%Construct dual system
ax=a1';
bx=c1';
p=[-2+2*sqrt(3)*i  -2-2*sqrt(3)*i];
k=acker(ax,bx,p);
% State observer gain
ke=k'
```

执行后得

```
The rank of observability matrix
ro=
      2
ke=
    -2
    17
```

例 3.37 离散系统

$$x(k+1) = gx(k) + hu(k)$$

其中

$$g = \begin{bmatrix} 0 & 1 \\ -0.16 & -1 \end{bmatrix} \quad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

设计反馈增益矩阵 k，使闭环系统极点为 0.5±j0.5。

解 MATLAB 程序为 ex3037.m：

```
% Example 3.37
%
% Pole placement in z plane
%
g=[0 1; -0.16 -1];
h=[0; 1];
disp('The rank of controllability matrix')
rc=rank(ctrb(g,h))
p=[0.5+0.5*i 0.5-0.5*i];
k=acker(g,h,p)
```

执行后得

```
The rank of controllability matrix
rc =
```

```
        2
k =
        0.3400    -2.0000
```

例 3.38 离散系统同 3.37，设计反馈增益矩阵 k，使系统具有无阻尼响应。

解 系统具有无阻尼响应，这时闭环极点为：$\mu_1=0$，$\mu_2=0$。因此 MATLAB 程序为 ex3038.m：

```
% Example 3.38
%
% Pole placement in z plane
%
g=[0 1; -0.16 -1];
h=[0; 1];
disp('The rank of controllability matrix')
rc=rank(ctrb(g,h))
p=[0 0];
k=acker(g,h,p)
```

执行后得

```
The rank of controllability matrix
rc =
        2
k =
     -0.1600    -1.0000
```

例 3.39 对例 3.33 中的倒立摆系统，设计数字控制器使闭环系统具有 $\mu_{1,2}=0.9\pm j0.25$，$\mu_3=\mu_4=\mu_5=0$ 的极点。

解 倒立摆系统模型为

$$\begin{cases}\dot{x} = ax + bu \\ y = cx\end{cases}$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 20.601 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -0.4905 & 0 & 0 & 0 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} \quad c = [0\ 0\ 1\ 0]$$

对这一连续时间模型进行离散化，可得倒立摆系统的离散时间模型（Ts=0.1 秒），这可由 c2d 函数实现：[g,h]=c2d(a,b,0.1)，这样就得到了离散时间模型

$$\begin{cases}x(k+1) = gx(k) + hu(k) \\ y(k) = cx(k)\end{cases}$$

由于系统本身不含积分环节，因此控制器中除状态反馈外还引入了一积分环节，如图 3.43 所示。

从图中可得到

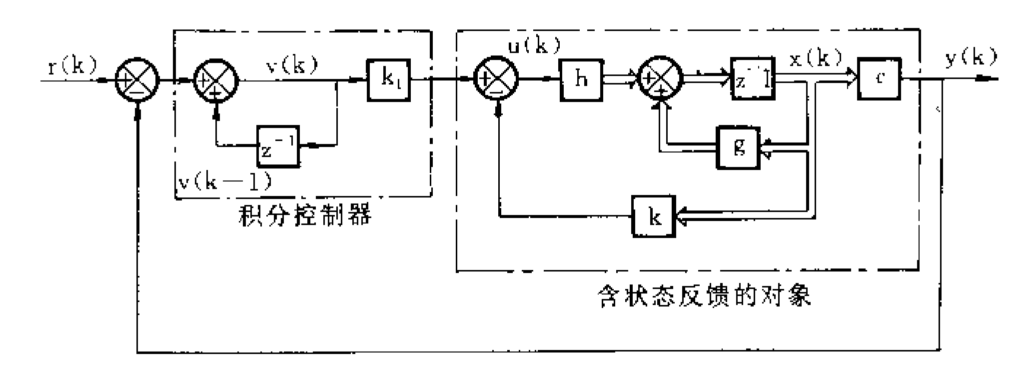


图 3.43 倒立摆系统的数字控制器

$$\begin{cases} v(k) = v(k-1) + r(k) - y(k) \\ v(k) = -kx(k) + k_I v(k) \end{cases}$$

令 $x_5(k)=v(k)$，$\xi(k)=[x_1(k),x_2(k),x_3(k),x_4(k),x_5(k)]'$，则可得到

$$\begin{cases} \xi(k+1) = \hat{g}\xi(k) + \hat{h}w(k) \\ w(k) = -\hat{k}\xi(k) \end{cases}$$

其中

$$\hat{g} = \begin{bmatrix} g & 0 \\ -cg & 1 \end{bmatrix} \quad \hat{h} = \begin{bmatrix} h \\ -ch \end{bmatrix}$$

$$\hat{k} = [k \ -k_I] = [k_1 \quad k_2 \quad k_3 \quad k_4 \quad -k_I]$$

这可由 MATLAB 的 acker 或 place 函数设计。MATLAB 程序为 ex3039.m：

```
% Example 3.39
%
% Design of an inverted pendulum control system
% using pole placement - - Ackermann
%
% Modelling
a=[0 1 0 0; 20.601 0 0 0; 0 0 0 1; -0.4905 0 0 0];
b=[0; -1; 0; 0.5];
c=[0 0 1 0];
d=[0];
% Discretization
[g,h]=c2d(a,b,0.1);
g1=[g zeros(4,1);-c * g 1];
h1=[h;-c * h];
% Check the controllability
disp('The rank of controllability matrix')
rc=rank(ctrb(g1,h1))

% Design
```

```
p=[0.9+0.25*i   0.9-0.25*i   0   0   0];
k=acker(g1,h1,p);
k1=k(1:4), ki=-k(5)
%
% Step response
%
% The close loop state system is denoted as (ac,bc,cc,dc)
%
gc=[g-h*k1 h*ki; -c*g+c*h*k1 1-c*h*ki];
hc=[0 0 0 0 1]';
cc=[c 0];
dc=[0];
figure(1)
[y,x]=dstep(gc,hc,cc,dc,1,100);
kk=1:length(y);
plot(kk,y,'o',kk,y,'-')
title('Step Response of inverted pendulum system')
xlabel('Samples')
ylabel('Position of Cart y=x3')
figure(2)
hold on
for j=1:5
y1=x(:,j);
plot(kk,y1)
end
grid on
title('Step Response Curves for x1,x2,x3,x4,x5')
xlabel('Samples')
ylabel('x1, x2, x3, x4, x5')
hold off
```

执行后得

```
The rank of controllability matrix
rc =

         5

k1 =

    -371.1822   -103.7521   -236.7336   -168.5108

ki =

    -72.6484
```

同时得到了系统的单位阶跃响应，如图 3.44 和图 3.45 所示。

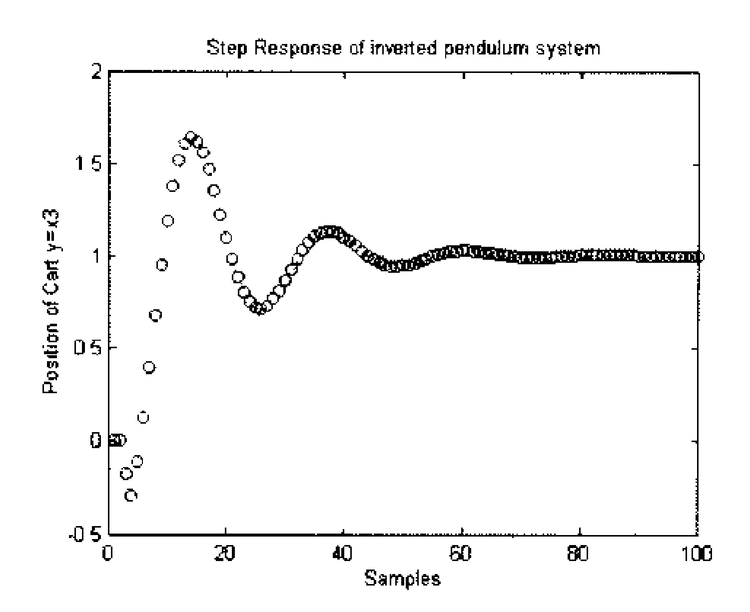


图 3.44 倒立摆小车位置的单位响应曲线

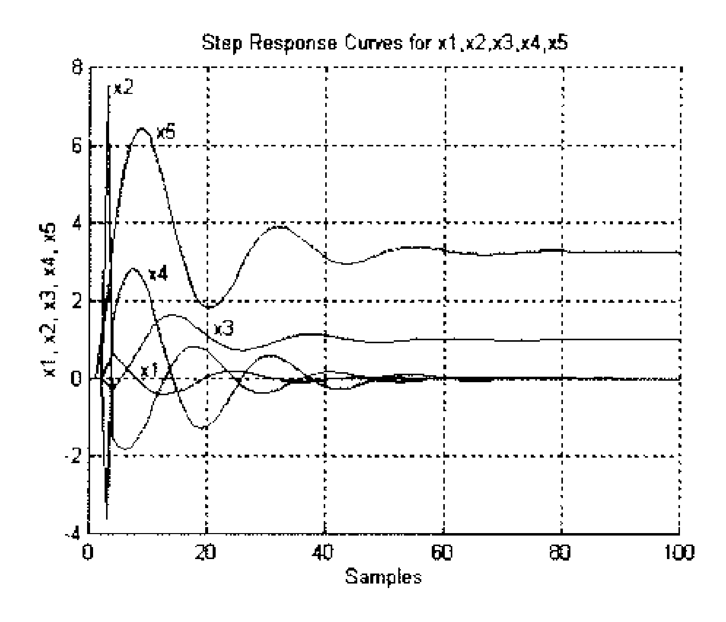


图 3.45 倒立摆状态的单位响应曲线

例 3.40 离散系统

$$\begin{cases} x(k+1) = gx(k) + hu(k) \\ y(k) = cx(k) \end{cases}$$

其中

$$g = \begin{bmatrix} 0 & -0.16 \\ 1 & -1 \end{bmatrix} \quad h = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad c = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

设计全阶状态观测器，其观测器的极点为 $\mu_{1,2} = 0.5 \pm j0.5$。

解 MATLAB 程序为 ex3040.m：

```
% Example 3.40
```

```
%
% Design of state observer
%
g=[0 -0.16; 1 -1];
h=[0; 1]; c=[0 1];
% Check the observability
disp('The rank of observability matrix')
ro=rank(obsv(g,c))
% Design
g1=g'; h1=c'; c1=h';
p=[0.5-0.5*i 0.5-0.5*i];
k=acker(g1,h1,p);
ke=k'
```

执行后得

```
The rank of observability matrix
ro =
        2
ke =
        0.3400
       -2.0000
```

例 3.41 离散系统

$$\begin{cases} x(k+1) = gx(k) + hu(k) \\ y(k) = cx(k) \end{cases}$$

其中

$$g = \begin{bmatrix} 0 & 0 & -0.25 \\ 1 & 0 & 0 \\ 0 & 1 & 0.5 \end{bmatrix} \qquad h = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \qquad c = [1 \quad 0 \quad 0]$$

设计最小阶状态观测器，使观测器具有无阻尼响应。

解 设计过程类似于例 3.33，MATLAB 程序为 ex3041.m：

```
% Example 3.41
%
% Design of state observer
%
g=[0 0 -0.25; 1 0 0; 0 1 0.5];
h=[1 0 1]'; c=[1 0 0];
% Check the observability
disp('The rank of observability matrix')
ro=rank(obsv(g,c))
```

```
% Design
gaa=g(1,1); gab=g(1,2:3);
gba=g(2:3,1); gbb=g(2:3,2:3);
ha=h(1,1); hb=h(2:3,1);
% Dual system
g1=gbb'; h1=gab';
p=[0 0];
k=acker(g1,h1,p);
ke=k'
```

执行后得

```
The rank of observability matrix
ro =
       3
ke =
       0
      -2
```

例 3.42 离散系统

$$\begin{cases} x(k+1) = gx(k) + hu(k) \\ y(k) = cx(k) \end{cases}$$

其中

$$g = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ -0.2 & -0.5 & 1.1 \end{bmatrix} \quad h = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad c = [1 \quad 0 \quad 0]$$

设计调节器使闭环系统具有无阻尼响应，并设计全阶状态观测器，使观测器也具有无阻尼响应。

解 MATLAB 程序为 ex3042.m：

```
% Example 3.42
%
% Design of state observer
%
g=[0 0 1; 1 0 0; -0.2 -0.5 1.1];
h=[0; 0; 1]; c=[1 0 0];
% Check the controllability and observability
disp('The rank of controllability matrix')
rc=rank(ctrb(g,h))
disp('The rank of observability matrix')
ro=rank(obsv(g,c))


% Design Digital controller
```

```
p=[0 0 0];
k=acker(g,h,p)

% Design state observer
gaa=g(1,1); gab=g(1,2:3);
gba=g(2:3,1); gbb=g(2:3,2:3);
ha=h(1,1); hb=h(2:3,1);
% Dual system
g1=gbb'; h1=gab';
p=[0 0];
k=acker(g1,h1,p);
ke=k'
```

执行后得

```
The rank of controllability matrix
rc =
     3
The rank of observability matrix
ro =
     3

k =
    -0.2000    -0.5000    1.1000
ke =
          0
     1.1000
```

3.6 最优控制器设计

例 3.43 系统模型

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$

性能指标为

$$J = \int_0^{\infty} (x^T Q x + u^T R u)\, dt$$

其中

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = [\ 1\ ]$$

设计反馈控制

$$u = -kx$$

使 J 最小。

解 可直接利用 MATLAB 的 lqr 函数进行设计。MATLAB 程序为 ex3043. m:

```
% Example 3.43
%
% Design of quadratic optimal regulator system
%
a=[0 1; 0 -1];
b=[0; 1];
q=[1 0; 0 1];
r=[1];
disp(' The optimal feedback gain matrix k is')
k=lqr(a,b,q,r)
```

执行后得

```
The optimal feedback gain matrix k is
k=
    1.0000    1.0000
```

例 3.44 考虑系统

$$\dot{x} = ax + bu$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -35 & -27 & -7 \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

性能指标为

$$J = \int_0^\infty (x^T Q x + u^T R u)\, dt$$

其中

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad R = [\,1\,]$$

设计最优控制器，并求出 Riccati 方程的解 p 及闭环系统 a－bk 的特征值(闭环系统的极点)。

解 这也可直接采用 lqr 函数，MATLAB 程序为 ex3044. m:

```
% Example 3.44
%
% Design of quadratic optimal regulator system
%
a=[0 1 0; 0 0 1; -35 -27 -9];
b=[0; 0; 1];
```

```
q=[1 0 0; 0 1 0; 0 0 1];
r=[1];
[k,p,e]=lqr(a,b,q,r);
disp(' The optimal feedback gain matrix k is')
k
disp(' The solution of Riccati equation is')
p
disp(' The eigenvalues of close-loop system are')
e
```

执行后得

```
The optimal feedback gain matrix k is
k =
    0.0143    0.1107    0.0676

The solution of Riccati equation is
p =
    4.2625    2.4957    0.0143
    2.4957    2.8150    0.1107
    0.0143    0.1107    0.0676

The eigenvalues of close-loop system are
e =
   -5.0958
   -1.9859 + 1.7110i
   -1.9859 - 1.7110i
```

这里得到的 p 满足代数 Riccati 方程

$$pa + a^Tp + Q - pbRb^Tp = 0$$

由此构成的闭环系统的三个极点均位于左半 s-平面，因而系统是稳定的。实际上，由最优控制构成的闭环系统都是稳定的，因为它是基于 Lyapunov 稳定性理论进行设计的。

例 3.45 系统模型

$$\dot{x} = \begin{bmatrix} -1 & 1 \\ 0 & 2 \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$

性能指标

$$J = \int_0^{\infty} (x^TQx + u^TRu)\, dt$$

其中

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = [\,1\,]$$

设计最优控制器，使 J 最小。

解 MATLAB 程序为 ex3045.m：

```
% Example 3.45
%
% Design of quadratic optimal regulator system
%
a=[-1 1; 0 2];
b=[1; 0];
q=[1 0; 0 1];
r=[1];
disp(' The optimal feedback gain matrix k is')
k=lqr(a,b,q,r)
```

执行后得

```
The optimal feedback gain matrix k is
Warning: Matrix is singular to working precision.

k =
     NaN    NaN
```

由于求得的 k 为非数值(NaN)，因此得不到最优控制器，事实上，设 $k=[k_1 \quad k_2]^T$，有闭环系统

$$a - bk = \begin{bmatrix} -1-k_1 & 1-k_2 \\ 0 & 2 \end{bmatrix}$$

$$|sI - a + bk| = (s+1+k_1)(s-2) = 0$$

因此闭环系统极点为

$$p_1 = -1-k_1 \qquad p_2 = 2$$

由于可见 $p_2=2$ 不可能由设计 k 来改变，而这是一个不稳定极点，因此无法得到最优控制器。

例 3.46 系统模型

$$\begin{cases} \dot{x} = ax + bu \\ y = cx + du \end{cases}$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & -2 & -3 \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad c = [1 \quad 0 \quad 0] \qquad d = [\,0\,]$$

设计最优控制器，使

$$J = \int_0^\infty (x^TQx + u^TRu)\,dt$$

最小，其中

$$Q = \begin{bmatrix} q_{11} & 0 & 0 \\ 0 & q_{22} & 0 \\ 0 & 0 & q_{33} \end{bmatrix} \qquad R = 0.01$$

解 最优控制结构如图 3.46 所示。

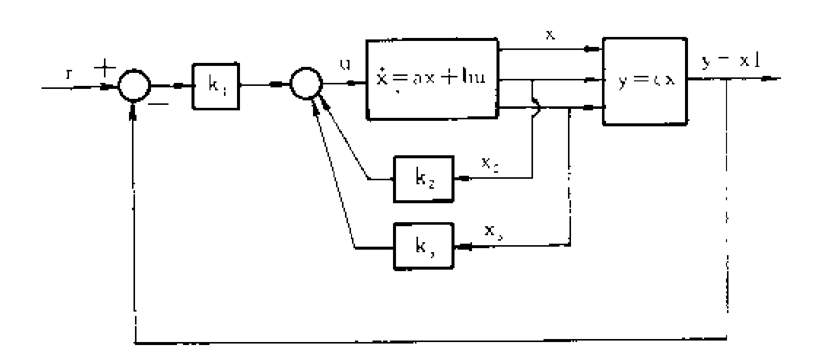


图 3.46 最优控制器结构

为使系统具有较快的响应，应选较大的 q_{11}，因此可选

$$q_{11}=100 \qquad q_{22}=q_{33}=1$$

这时 MATLAB 程序为 ex3046.m：

```
% Example 3.46
%
% Design of quadratic optimal regulator system
%
a=[0 1 0; 0 0 1; 0 -2 -3];
b=[0; 0; 1];
c=[1 0 0];
d=[0];
q=[100 0 0; 0 1 0; 0 0 1];
r=[0.01];
[k,p,e]=lqr(a,b,q,r);
disp(' The optimal feedback gain matrix k is')
k
%
% Step response
%
% The close loop state system is denoted as (ac,bc,cc,dc)
%
k1=k(1);
ac=a-b*k;
bc=b*k1;
cc=c; dc=d;
figure(1)
step(ac,bc,cc,dc)
title('Step Response of Quadratic Optimal Control system')
```

```
xlabel('Sec')
ylabel('Output y=x1')
figure(2)
[y,x,t]=step(ac,bc,cc,dc);
plot(t,x,'y')
title('Step Response Curves for x1,x2,x3')
xlabel('Sec')
ylabel('x1, x2, x3')
```

执行后得

```
The optimal feedback gain matrix k is
k=
    100.0000    53.1200    11.6711
```

同时还得到了如图 3.47 所示的单位阶跃输出响应、图 3.48 所示的状态响应和图 3.49 所示的控制信号。

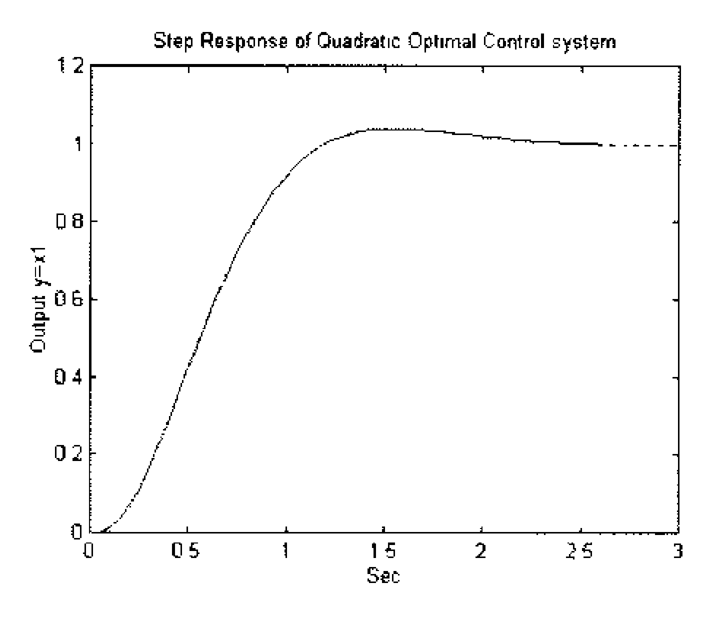


图 3.47 最优控制系统输出响应($q_{11}=100$)

为研究 q 阵对最优控制系统设计的影响，现改 $q_{11}=1$，其余参数不变，执行后得

```
k=
    10.0000    16.5022    8.9166
```

同时也产生了相应的结果曲线，如图 3.50～3.52 所示。

比较图 3.47 与图 3.50 可知，由于 Q 阵不同，系统输出响应有较大差异，这是因为输出仅与 x_1 有关，因此在指标中加大 x_1 的权值，表示控制 u 对 x_1 的作用增强，因此输出建立时间短。

比较图 3.49 与图 3.52 可知，系统控制输入在两种情况下形状类似，但其幅度差异甚远，这也是由于指标加权阵 Q 不同之故。

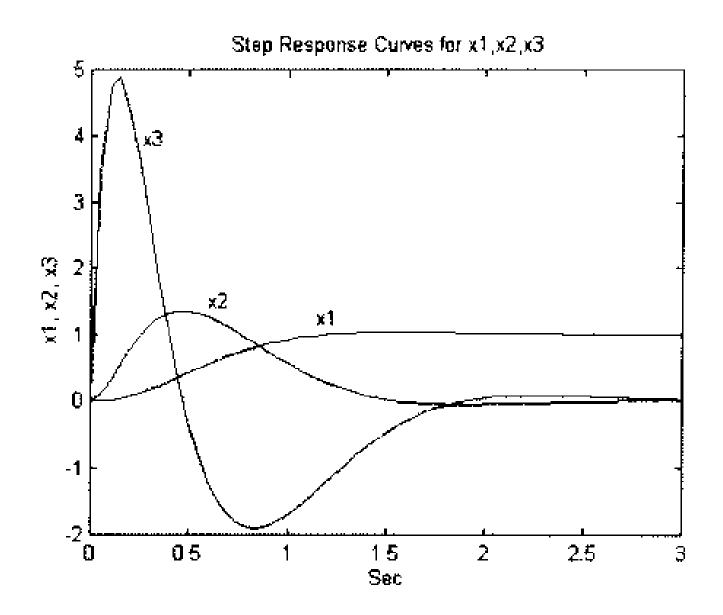


图 3.48　最优控制系统状态响应($q_{11}=100$)

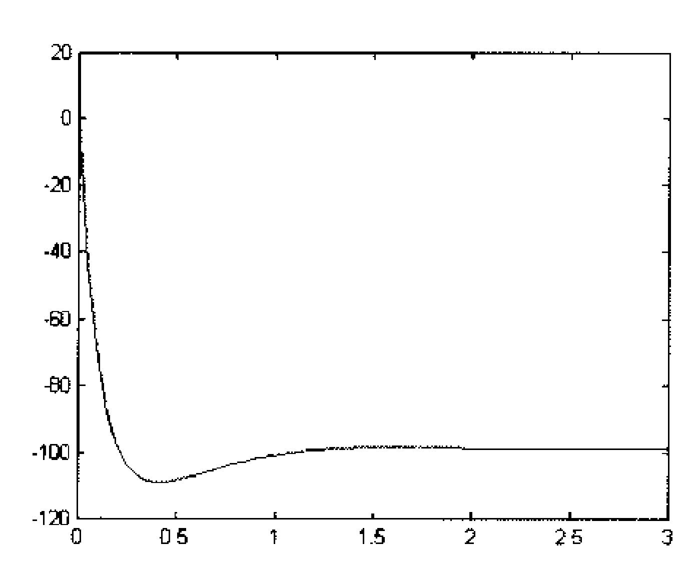


图 3.49　最优控制系统控制信号($q_{11}=100$)

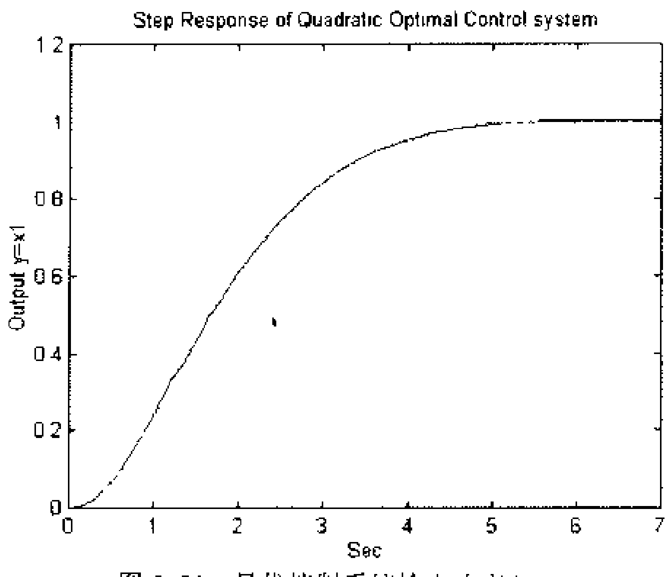


图 3.50　最优控制系统输出响应($q_{11}=1$)

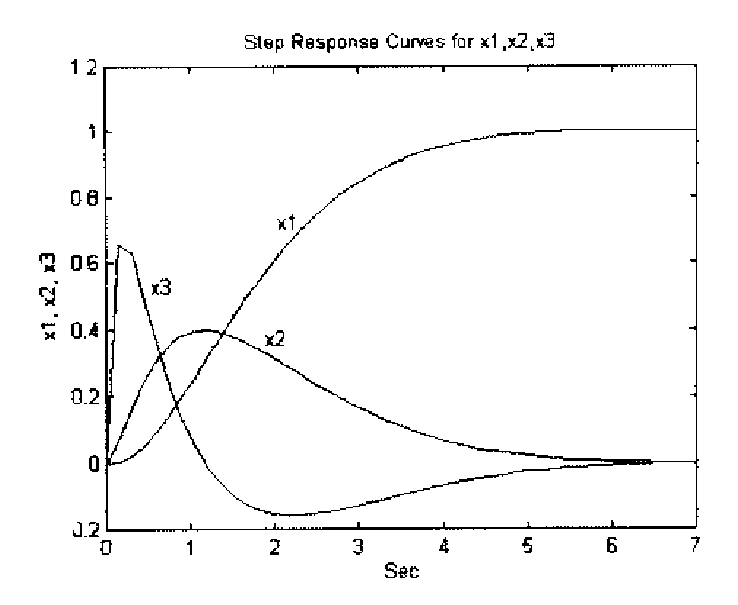


图 3.51　最优控制系统状态响应($q_{11}=1$)

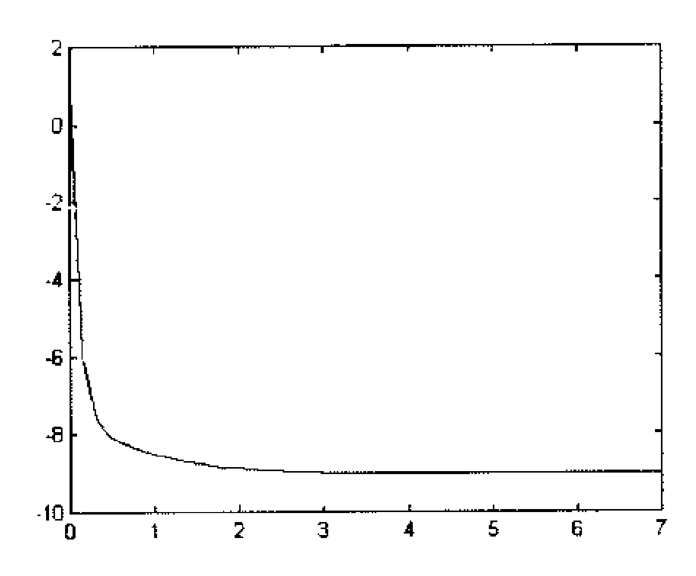


图 3.52　最优控制系统控制信号($q_{11}=1$)

例 3.47　考虑系统

$$\begin{cases}\dot{x}(t)=\begin{bmatrix}-0.2 & 0.5 & 0 & 0 & 0\\ 0 & -0.5 & 1.6 & 0 & 0\\ 0 & 0 & -14.3 & 85.8 & 0\\ 0 & 0 & 0 & -33.3 & 100\\ 0 & 0 & 0 & 0 & -10\end{bmatrix}x(t)+\begin{bmatrix}0\\0\\0\\0\\30\end{bmatrix}u(t)\\ y(t)=[1\ 0\ 0\ 0\ 0]\,x(t)\end{cases}$$

设计最优控制器，使性能指标

$$J=\int_0^{\infty}(x^TQx+u^TRu)\,dt$$

最小，其中

$$Q = \mathrm{diag}\{1,1,1,1,1\} \qquad R = 1$$

解 diag 函数可产生对角阵，详见《MATLAB 程序设计语言》(西电)。为给出控制效果，绘出了闭环系统的单位阶跃输出和状态响应。MATLAB 程序为 ex3047.m：

```
% Example 3.47
%
% Design of quadratic optimal regulator system
%
a=-diag([0.2 0.5 14.3 33.3 10])+diag([0.5 1.6 85.8 100],1);
b=[0 0 0 0 30]';
c=[1 0 0 0 0]; d=[0];
q=diag([1 1 1 1 1]); r=1;
[k,p,e]=lqr(a,b,q,r);
disp(' The optimal feedback gain matrix k is')
k
%
% Step response
%
% The close loop state system is denoted as (ac,bc,cc,dc)
%
k1=k(1);
ac=a-b*k;
bc=b*k1;
cc=c; dc=d;
figure(1)
step(ac,bc,cc,dc)
title('Step Response of Quadratic Optimal Control system')
xlabel('Sec')
ylabel('Output y=x1')
figure(2)
[y,x,t]=step(ac,bc,cc,dc);
plot(t,x,'y')
title('Step Response Curves for x1,x2,x3,x4,x5')
xlabel('Sec')
ylabel('x1,x2,x3,x4,x5')
```

执行后得

```
  The optimal feedback gain matrix k is
k =
     0.5936     0.8940     0.6441     1.4422     2.9417
```

同时得到了如图 3.53 所示的输出阶跃响应和如图 3.54 所示的状态阶跃响应。

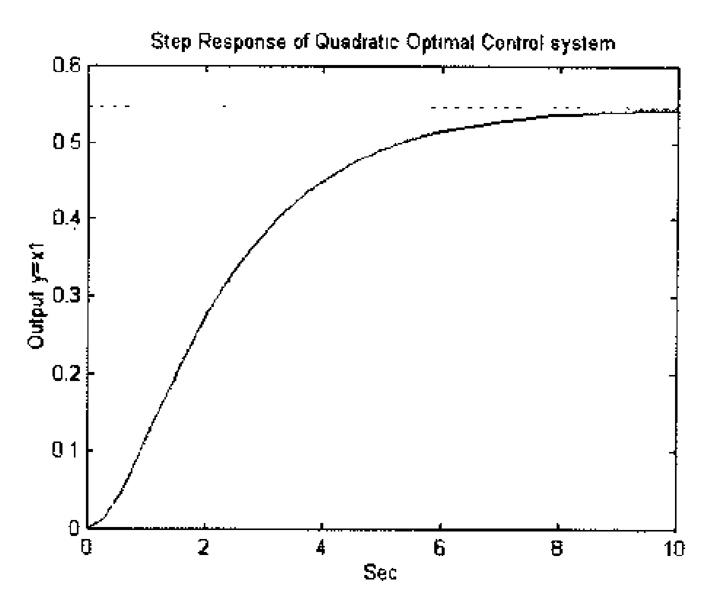


图 3.53　闭环系统的输出单位阶跃响应

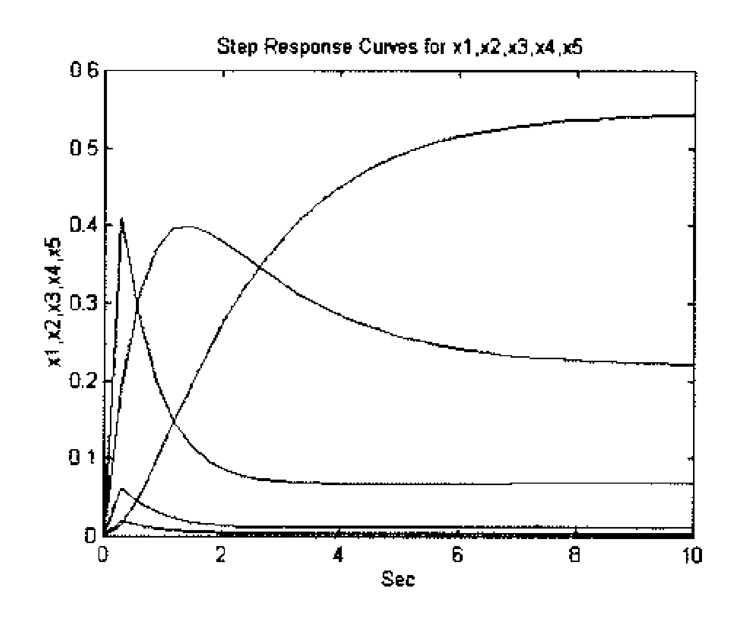


图 3.54　闭环系统的状态单位阶跃响应

例 3.48　考虑与例 3.47 具有相同状态方程的系统，其输出方程为

$$y(t) = [1\ 1\ 1\ 1\ 1]\, x(t)$$

现要求采用输出反馈，即设计 $u=-ky$，使性能指标

$$J = \int_0^{\infty} (y^T Q y + u^T R u)\, dt$$

最小，其中 $Q=1$，$R=1$。

解　特别注意，这里不能直接利用 lqry 函数进行设计。下面我们给出以 MATLAB 基本函数求取最优控制的程序，MATLAB 程序为ex3048.m：

```
% Example 3.48
%
```

```
% Design of quadratic optimal regulator system
%
a=-diag([0.2 0.5 14.3 33.3 10])+diag([0.5 1.6 85.8 100],1);
b=[0 0 0 0 30]';
c=[1 0 4 3 2]; d=[0];
q=diag([1,1,1,1,1]); r=1;

% Design with basic functions of MATLAB
tol=1e-10;
k1=1;
I=eye(size(a));
while(1)
  a0=a-b*k1*c;
  p=lyap(a0',c'*k1*r*k1*c+q);
  z=lyap(a0,I);
  k0=inv(r)*b'*p*z*c'*inv(c*z*c');
  if (norm(k0-k1,1)>tol), k1=k0; else break; end
end
disp(' The optimal feedback gain matrix k is')
k=k0
%
% Step response
%
% The close loop state system is denoted as (ac,bc,cc,dc)
%
ac=a-b*k*c;
bc=b;
cc=c; dc=d;
figure(1)
step(ac,bc,cc,dc)
title('Step Response of Quadratic Optimal Control system')
xlabel('Sec')
ylabel('Output y(t)')
axis([0 0.3 0 1])
```

执行后得最优控制增益阵

```
The optimal feedback gain matrix k is
k=
   1.6788
```

同时还得到了这一闭环系统的输出单位阶跃响应，如图 3.55 所示。

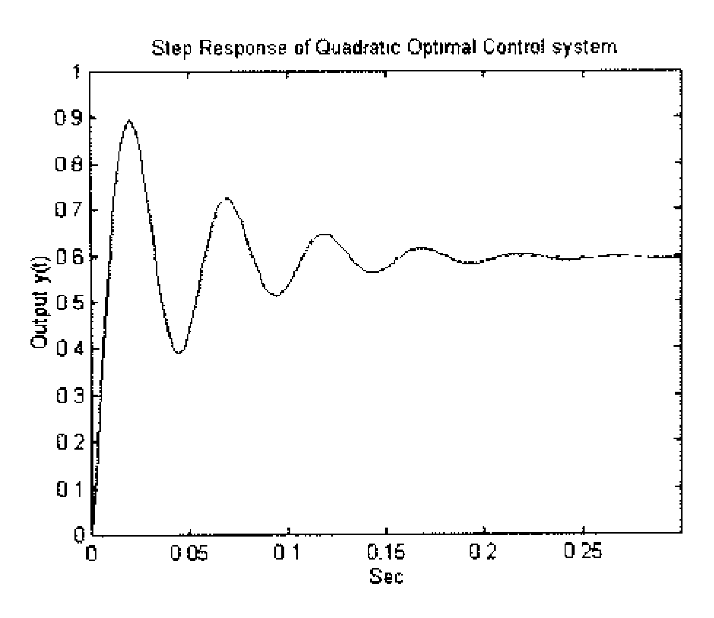


图 3.55 闭环系统输出单位阶跃响应

例 3.49 考虑与例 3.47 具有相同状态方程，其输出方程为

$$y(t) = [1\ 1\ 1\ 1\ 1]\, x(t)$$

仍以状态反馈构成控制律 $u=-kx$，使性能指标

$$J = \int_0^\infty (y^T Q y + u^T R u)\, dt$$

最小，其中 Q=1，R=1。

解 这可利用控制工具箱函数 lqry 进行设计，因此 MATLAB 程序为 ex3049.m：

```
% Example 3.49
%
% Design of quadratic optimal regulator system
%
a=-diag([0.2 0.5 14.3 33.3 10])+diag([0.5 1.6 85.8 100],1);
b=[0 0 0 0 30]';
c=[1 0 0 0 0]; d=[0];
q=diag([1]); r=1;
k=lqry(a,b,c,d,q,r);
disp(' The optimal feedback gain matrix k is')
k
%
% Step response
%
% The close loop state system is denoted as (ac,bc,cc,dc)
%
k1=k(1);
```

```
ac=a-b*k;
bc=b*k1;
cc=c; dc=d;
figure(1)
step(ac,bc,cc,dc)
title('Step Response of Quadratic Optimal Control system')
xlabel('Sec')
ylabel('Output y=x1')
figure(2)
[y,x,t]=step(ac,bc,cc,dc);
plot(t,x,'y')
title('Step Response Curves for x1,x2,x3,x4,x5')
xlabel('Sec')
ylabel('x1,x2,x3,x4,x5')
```

执行后得

```
The optimal feedback gain matrix k is
k =
     0.9260    0.1678    0.0157    0.0371    0.2653
```

同时得到了如图 3.56 和 3.57 所示的输出和状态单位阶跃响应。

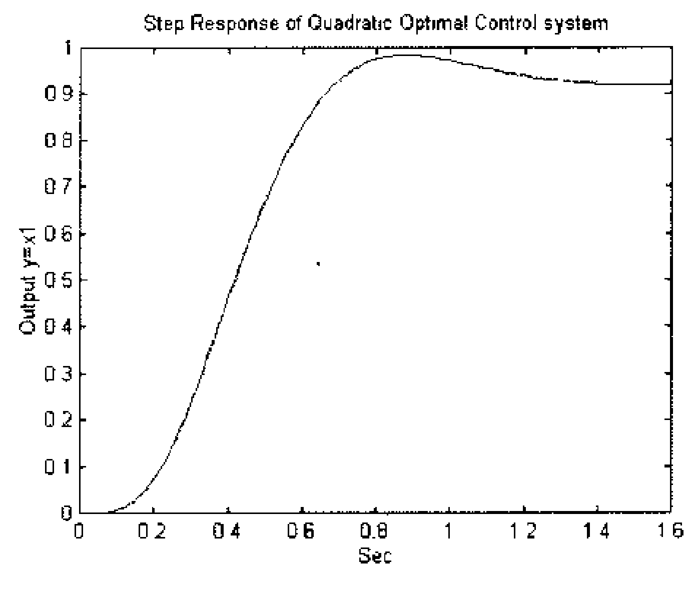


图 3.56　闭环系统输出单位阶跃响应

比较图 3.53 和图 3.56 可以得到，由于采用性能指标的不同，致使闭环系统输出响应建立时间相差甚远。在例 3.49 中，由于采用输出作为性能指标，使输出具有较快的响应，而在例 3.47 中，采用状态作为性能指标，且 $Q=\mathrm{diag}([1,1,1,1,1])$，即对所有状态作同等重要看待，而 $y(t)=cx(t)=x_1(t)$，因此对输出没有体现至关重要这一特性，故在闭环系统输出响应中，输出响应较慢。读者不妨修改 Q 阵，使 $q_{11}\gg\max(q_{22},q_{33},q_{44},q_{55})$，这样从例 3.47中可得到与例 3.49 中极为相近的输出响应，当 $q_{11}=1$，$q_{22}=q_{33}=q_{44}=q_{55}=0$ 时，两者

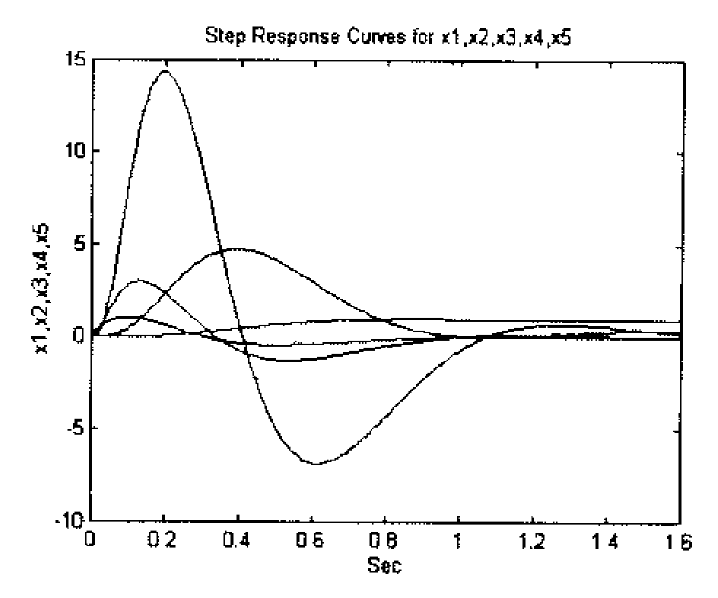


图 3.57　闭环系统状态单位阶跃响应

具有相同的输出响应。

例 3.50　考虑离散系统

$$x(k+1) = gx(k) + hu(k)$$

其中

$$g = \begin{bmatrix} 0.2 & 0 \\ 0 & 0.4 \end{bmatrix} \qquad h = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

设计最优控制器，使性能指标

$$J = \frac{1}{2}\sum_{k=0}^{\infty}[x^T(k)Qx(k) + u^T(k)Ru(k)]$$

最小，其中

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix} \qquad R = 1$$

解　控制律为 $u(k) = -kx(k)$，现利用 MATLAB 控制工具箱函数 dlqr 来得到反馈增益 k。MATLAB 程序为 ex3050.m：

```
% Example 3.50
%
% Design of quadratic optimal regulator system
%
g=[0.2 0; 0 0.4];
h=[1; 1];
q=[1 0; 0 0.5];
r=[1];
[k,p,e]=dlqr(g,h,q,r);
disp('The optimal feedback gain matrix k is')
k
```

执行后得

```
The optimal feedback gain matrix k is
k =
    0.0786    0.0865
```

例 3.51 不含积分环节的伺服系统设计。伺服系统与控制器结构如图 3.58 所示，k_1，k_2 为控制器参数，要求设计最优控制器使 J 最小。

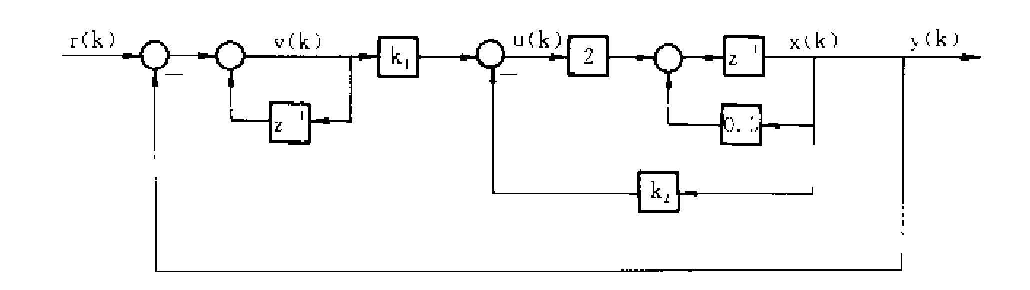


图 3.58 伺服系统最优控制器结构

解 由图 3.58 可得

$$\begin{cases} x(k+1) = 0.5x(k) + 2u(k) \\ u(k) = k_1 v(k) - k_2 x(k) \\ v(k) = r(k) - y(k) + v(k-1) \\ y(k) = x(k) \end{cases}$$

因此得系统方程

$$\begin{cases} x(k+1) = gx(k) + hw(k) \\ w(k) = -kx(k) \end{cases}$$

其中

$$x(k) = [x_1(k) \quad x_2(k)]^T, \quad k = [k_2 \quad -k_1]$$

$$g = \begin{bmatrix} 0.5 & 0 \\ -0.5 & 1 \end{bmatrix} \qquad h = \begin{bmatrix} 2 \\ -2 \end{bmatrix}$$

现设

$$J = \frac{1}{2}\sum_{k=0}^{\infty}[x^T(k)Qx(k) + w^T(k)Rw(k)]$$

并取

$$Q = \begin{bmatrix} 100 & 0 \\ 0 & 1 \end{bmatrix} \qquad R = 1$$

这样就可利用 MATLAB 的 dlqr 函数求出 k，进而得到 k_1,k_2，最后得到闭环系统的单位阶跃响应。

MATLAB 程序为 ex3051.m：

```
% Example 3.51
%
% Design of quadratic optimal regulator system
%
```

```
g=[0.5 0; -0.5 1];
h=[2; -2];
q=[100 0; 0 1];
r=[1];
[k,p,e]=dlqr(g,h,q,r);
disp('The optimal feedback gain matrix k is')
k1=-k(2)
k2=k(1)
%
% Step response
%
% The close loop state system is denoted as (gc,hc,cc,dc)
%
gc=g-h*k;
hc=[0; 1];
cc=[1 0]; dc=[0];
figure(1)
v=[0 100 0 1.2];
[y,x]=dstep(gc,hc,cc,dc,1,100);
kk=1:length(y);
plot(kk,y,'oy',kk,y,'-w')
axis(v)
title('Step Response of Servo Control system')
xlabel('Sample')
ylabel('Output y=x1')
figure(2)
plot(kk,x,'oy',kk,x,'-w')
title('Step Response Curves for x1,x2')
xlabel('Sample')
ylabel('x1, x2')
```

执行后得

```
The optimal feedback gain matrix k is
k1 =
      0.0475
k2 =
      0.2494
```

同时还得到了闭环系统的输出单位阶跃响应和状态单位阶跃响应，如图 3.59 和 3.60 所示。

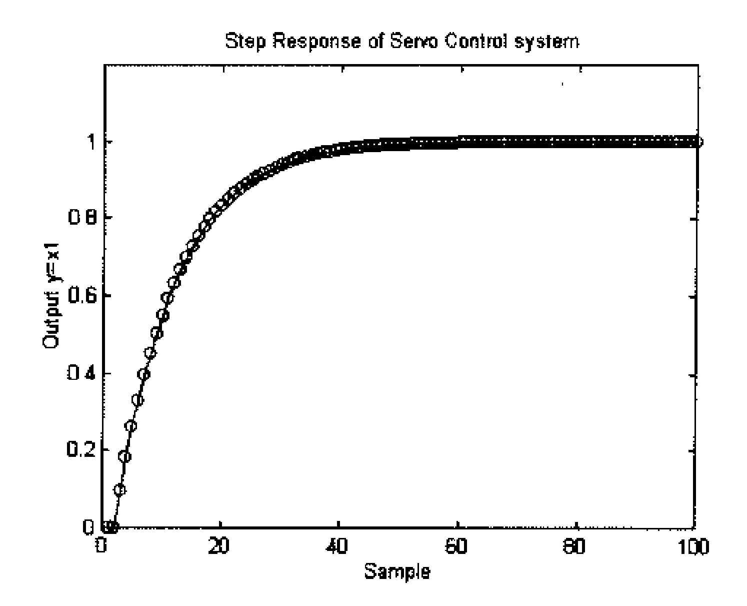


图 3.59 伺服系统输出单位阶跃响应

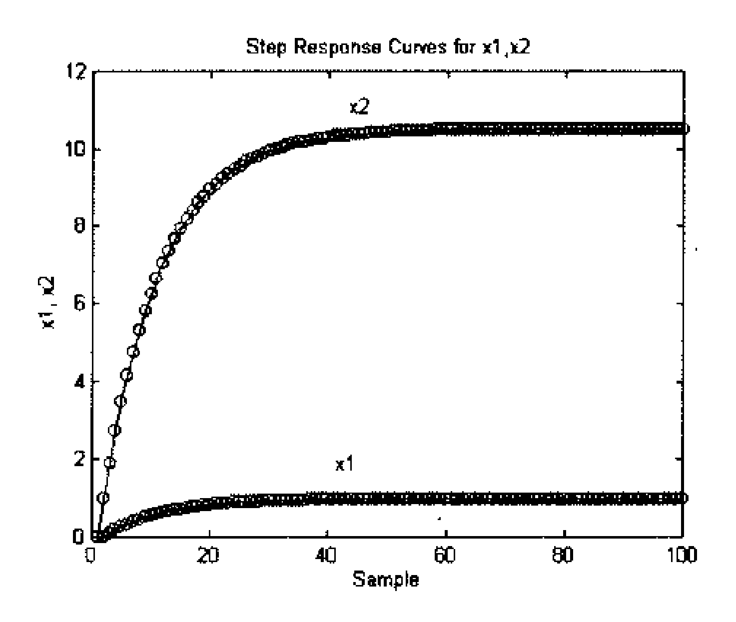


图 3.60 伺服系统状态单位阶跃响应

例 3.52 设计倒立摆系统的数字最优控制器。

解 倒立摆系统的建模可参见例 3.33，由此得连续系统

$$\begin{cases} \dot{x} = ax + bu \\ y = cx + du \end{cases}$$

其中

$$a = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 20.601 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ -0.4905 & 0 & 0 & 0 \end{bmatrix} \qquad b = \begin{bmatrix} 0 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} \qquad c = [0\ 0\ 1\ 0] \qquad d = 0$$

按 T=0.1s 进行取样离散化，可得离散系统模型(可利用 MATLAB 的 c2d 函数进行离散化)

$$\begin{cases} x(k+1) = gx(k) + hu(k) \\ y(k) = cx(k) - du(k) \end{cases}$$

其中

$$g = \begin{bmatrix} 1.1048 & 0.1035 & 0 & 0 \\ 2.1316 & 1.1048 & 0 & 0 \\ -0.0025 & -0.0001 & 1 & 0.1 \\ -0.0508 & -0.0025 & 0 & 1 \end{bmatrix} \qquad h = \begin{bmatrix} -0.0051 \\ -0.1035 \\ 0.0025 \\ 0.0501 \end{bmatrix}$$

由于系统中不含积分环节，因此在控制器中引入积分器，控制器结构如图 3.61 所示。

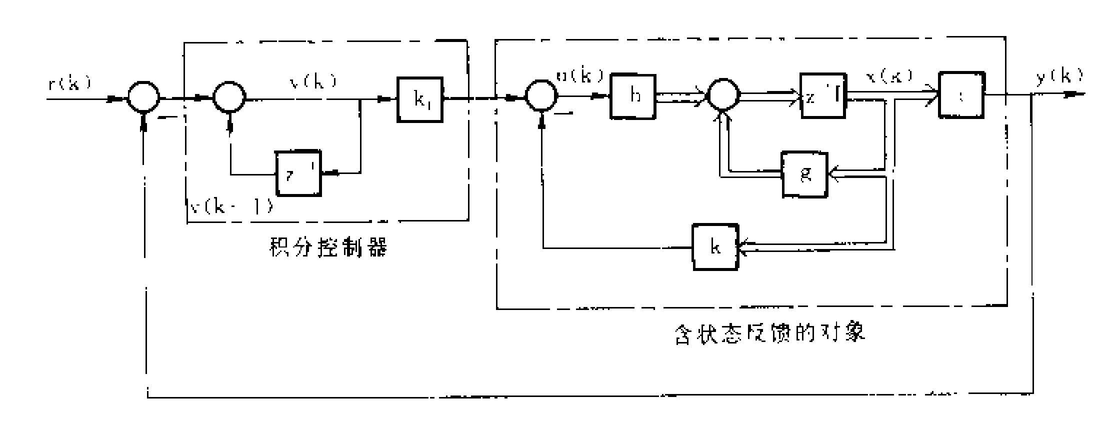


图 3.61　倒立摆最优控制系统

由图 3.61 可以得到增广后的状态方程

$$\begin{cases} p(k+1) = g_1 p(k) + h_1 w(k) \\ w(k) = -k_1 p(k) \end{cases}$$

其中

$$g_1 = \begin{bmatrix} g & 0 \\ -cg & 1 \end{bmatrix} \qquad h_1 = \begin{bmatrix} h \\ -ch \end{bmatrix} \qquad k_1 = [k \quad -k_I]$$

因此问题就变成，设计 k_1 使

$$J = \frac{1}{2}\sum_{k=0}^{\infty}[p^T Q p + w^T R w]$$

最小，这里 Q=diag([10,1,100,1,1])，R=[1]，也就是我们着重于 $x_1(k)$和 $x_3(k)$，特别是 $x_3(k)$(位置)是控制的最终目标。

MATLAB 程序为 ex3052.m：

```
% Example 3.52
%
% Design of an inverted pendulum control system
% using optimal regulator
%
% Design
a=[0 1 0 0; 20.601 0 0 0; 0 0 0 1; -0.4905 0 0 0];
```

```
b=[0; -1; 0; 0.5];
c=[0 0 1 0];
d=[0];
[g,h]=c2d(a,b,0.1)
g1=[g,zeros(4,1); -c*g, 1];
h1=[h; -c*h];
q=diag([10,1,100,1,1]); r=[1];
[k1,p,e]=dlqr(g1,h1,q,r);
disp('The optimal feedback gain matrix k is')
ki=-k1(1,5)
k=k1(1,1:4)
%
% Step response
%
% The close loop state system is denoted as (gc,hc,cc,dc)
%
gc=g1-h1*k1;
hc=[0; 0; 0; 0; 1];
cc=[0 0 1 0 0]; dc=[0];
figure(1)
v=[0 100 0 1.2];
[y,x]=dstep(gc,hc,cc,dc,1,100);
kk=1:length(y);
plot(kk,y,'oy',kk,y,'-w')
axis(v)
title('Step Response of Servo Control system')
xlabel('Sample')
ylabel('Output y(position of cart)')
figure(2)
v=[0 100 -0.2 0.8];
subplot(2,2,1)
ww=[1,0,0,0,0]; y1=x*ww';
plot(kk,y1,'oy',kk,y1,'-w'); axis(v)
title('Amgular Displacement Theta'); ylabel('x1')
subplot(2,2,2)
ww=[0,1,0,0,0]; y1=x*ww';
plot(kk,y1,'oy',kk,y1,'-w'); axis(v)
title('Amgular Velocity Theta Dot'); ylabel('x2')
subplot(2,2,3)
```

```
ww=[0,0,0,1,0]; y1=x*ww';
plot(kk,y1,'oy',kk,y1,'-w'); axis(v)
title('Velocity of Cart'); xlabel('Sample'); ylabel('x4')
v=[0 100 -5 25];
subplot(2,2,4)
ww=[0,0,0,0,1]; y1=x*ww';
plot(kk,y1,'oy',kk,y1,'-w'); axis(v)
title('Output of Integrator'); xlabel('Sample'); ylabel('x5')
```

执行后得

```
g =
     1.1048     0.1035          0          0
     2.1316     1.1048          0          0
    -0.0025    -0.0001     1.0000     0.1000
    -0.0508    -0.0025          0     1.0000
h =
    -0.0051
    -0.1035
     0.0025
     0.0501

The optimal feedback gain matrix k is
ki =
    -0.5189
k =
    -64.9795   -14.4902   -10.8487   -9.2883
```

同时还得到了输出及状态的单位阶跃响应，如图 3.62 和 3.63 所示。

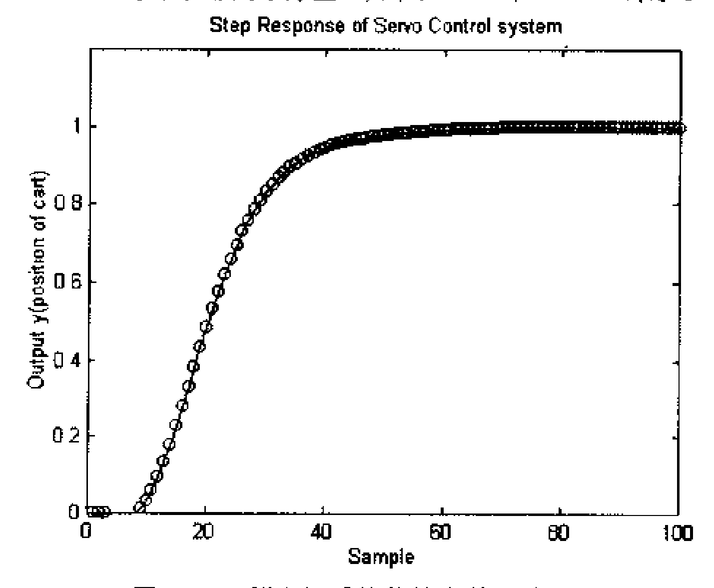


图 3.62　倒立摆系统的输出单位阶跃响应

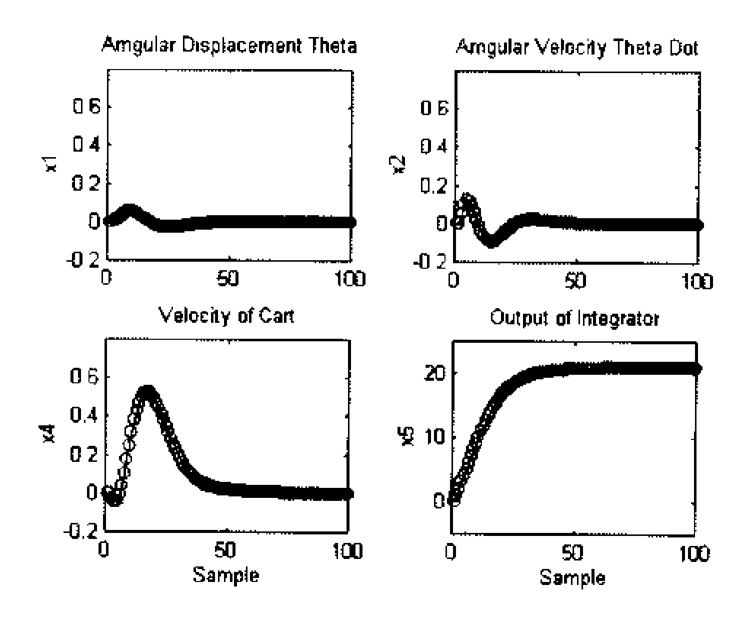


图 3.63　倒立摆系统的状态单位阶跃响应



MATLAB 命令参考

MATLAB 系统提供近 20 类基本命令函数，它们有一部分是 MATLAB 的内部命令，有一部分是以 M 文件形式出现的函数，这些 M 文件按类归于一子目录下，每个目录中除了以 M 文件表示的函数命令之外，还有一个特殊的文件 contents.m，它包含了该目录各个 M 文件的简介。每个函数文件中都包含了这一函数的用法指南，因此可用命令：

help fn

来显示有关函数 fn 的帮助信息(fn 为 M 文件名)，也可用命令：

help dn

来显示该目录下各函数文件的简要说明(dn 为目录名)。

限于篇幅，本附录不再列出各个函数详细说明，用户可利用 help 命令获得这些信息，也可参看《MATLAB 程序设计语言》一书。

表 A.1 为 20 类基本命令函数的子目录及其含义，表 A.2～A.20 中列出各类函数的简要说明，以供用户参考。

表 A.1　基本命令函数目录

目录名	命令函数	索引
general	通用命令	表 A.2
ops	操作符和特殊字符	表 A.3
elfun	基本数学函数	表 A.4
specfun	特殊数学函数	表 A.5
elmat	基本矩阵和矩阵操作	表 A.6
specmat	特殊矩阵	表 A.7
matfun	矩阵函数——数值线性代数	表 A.8
sparfun	稀疏矩阵函数	表 A.9
datafun	数据分析和傅里叶变换函数	表 A.10
funfun	泛函——非线性数值方法	表 A.11
polyfun	多项式和内插函数	表 A.12
graphics	通用图形函数	表 A.13
plotxy	二维图形函数	表 A.14
plotxyz	三维图形函数	表 A.15

续表

目录名	命令函数	索引
lang	语言结构和调试	表 A.16
color	颜色控制和亮度模型函数	表 A.17
strfun	字符串函数	表 A.18
sounds	音频处理函数	表 A.19
iofun	低级文件 I/O 函数	表 A.20
demos	演示例子	—

表 A.2 通用命令

■ 管理命令和函数		
	help	在线帮助文本
	doc	装入超文本说明
	what	M、MAT、MEX 文件的目录列表
	type	列出 M 文件
	lookfor	通过 help 条目搜索关键字
	which	定位函数和文件
	demo	运行演示程序
	path	控制 MATLAB 的搜索路径
■ 管理变量和工作空间		
	who	列出当前变量
	whos	列出当前变量(长表)
	load	从磁盘文件中恢复变量
	save	保存工作空间变量
	clear	从内存中清除变量和函数
	pack	整理工作空间内存
	size	矩阵的尺寸
	length	向量的长度
	disp	显示矩阵或文本
■ 与文件和操作系统有关的命令		
	cd	改变当前工作目录
	dir	目录列表
	delete	删除文件

续表

■ 与文件和操作系统有关的命令		
	getenv	获取环境变量值
	!	执行操作系统命令
	unix	执行操作系统命令并返回结果
	diary	保存 MATLAB 任务
■ 控制命令窗口		
	cedit	设置命令行编辑
	clc	清命令窗口
	home	光标置左上角
	format	设置输出格式
	echo	MATLAB 文件内使用的回显命令
	more	在命令窗口中控制分页输出
■ 启动和退出 MATLAB		
	quit	退出 MATLAB
	startup	引用 MATLAB 时所执行的 M 文件
	matlabrc	主启动 M 文件
■ 一般信息		
	info	MATLAB 系统信息及 Mathworks 公司信息
	subscribe	成为 MATLAB 的订购用户
	hostid	MATLAB 主服务程序的识别代号
	whatsnew	在说明书中未包含的新信息
	ver	版本信息

表 A.3 操作符和特殊字符

■ 操作符和特殊字符		
	+	加
	−	减
	*	矩阵乘法
	.*	数组乘法
	^	矩阵幂
	.^	数组幂
	\	左除或反斜杠

续表

<table>
<tr><td colspan="3">■ 操作符和特殊字符</td></tr>
<tr><td></td><td>/</td><td>右除或斜杠</td></tr>
<tr><td></td><td>./</td><td>数组除</td></tr>
<tr><td></td><td>kron</td><td>Kronecker 张量积</td></tr>
<tr><td></td><td>:</td><td>冒号</td></tr>
<tr><td></td><td>()</td><td>圆括号</td></tr>
<tr><td></td><td>[]</td><td>方括号</td></tr>
<tr><td></td><td>.</td><td>小数点</td></tr>
<tr><td></td><td>..</td><td>父目录</td></tr>
<tr><td></td><td>...</td><td>继续</td></tr>
<tr><td></td><td>,</td><td>逗号</td></tr>
<tr><td></td><td>;</td><td>分号</td></tr>
<tr><td></td><td>%</td><td>注释</td></tr>
<tr><td></td><td>!</td><td>感叹号</td></tr>
<tr><td></td><td>'</td><td>转置或引用</td></tr>
<tr><td></td><td>=</td><td>赋值</td></tr>
<tr><td></td><td>==</td><td>相等</td></tr>
<tr><td></td><td>< ></td><td>关系操作符</td></tr>
<tr><td></td><td>&</td><td>逻辑与</td></tr>
<tr><td></td><td>|</td><td>逻辑或</td></tr>
<tr><td></td><td>~</td><td>逻辑非</td></tr>
<tr><td></td><td>xor</td><td>逻辑异或</td></tr>
<tr><td colspan="3">■ 逻辑函数</td></tr>
<tr><td></td><td>exist</td><td>检查变量或函数是否存在</td></tr>
<tr><td></td><td>any</td><td>向量的任一元为真，则其值为真</td></tr>
<tr><td></td><td>all</td><td>向量的所有元为真，则其值为真</td></tr>
<tr><td></td><td>find</td><td>找出非零元素的索引号</td></tr>
<tr><td></td><td>isnan</td><td>当含 NaN 时，其值为真</td></tr>
<tr><td></td><td>isinf</td><td>当含无限大元时，其值为真</td></tr>
<tr><td></td><td>finite</td><td>当含有限值元时，其值为真</td></tr>
<tr><td></td><td>isempt</td><td>当矩阵为空矩阵时，其值为真</td></tr>
<tr><td></td><td>isreal</td><td>当矩阵为实矩阵时，其值为真</td></tr>
<tr><td></td><td>issparse</td><td>当矩阵为稀疏矩阵时，其值为真</td></tr>
<tr><td></td><td>isstr</td><td>当矩阵为文本串时，其值为真</td></tr>
<tr><td></td><td>isglobal</td><td>当变量为全局变量时，其值为真</td></tr>
</table>

表 A.4 基本数学函数

■ 三角函数	
sin	正弦
sinh	双曲正弦
asin	反正弦
asinh	反双曲正弦
cos	余弦
cosh	双曲余弦
acos	反余弦
acosh	反双曲余弦
tan	正切
tanh	双曲正切
atan	反正切
atan2	四象限反正切
atanh	反双曲正切
sec	正割
sech	双曲正割
asec	反正割
asech	反双曲正割
csc	余割
csch	双曲余割
acsc	反余割
acsch	反双曲余割
cot	余切
coth	双曲余切
acot	反余切
acoth	反双曲余切
■ 指数函数	
exp	指数
log	自然对数
log10	常用对数
sqrt	平方根

续表

■ 复数函数	
abs	绝对值
angle	相角
conj	复共轭
image	复数虚部
real	复数实部
■ 数值函数	
fix	朝零方向取整
floor	朝负无穷大方向取整
ceil	朝正无穷大方向取整
round	朝最近的整数取整
rem	除后余数
sign	符号函数

表 A.5 特殊数学函数

besselj	第一类 Bessel(贝塞尔)函数
bessely	第二类 Bessel 函数
besseli	改进的第一类 Bessel 函数
besselk	改进的第二类 Bessel 函数
beta	β 类函数
betainc	非完全的 β 函数
betaln	β 函数的对数
ellipj	雅可比椭圆函数
ellipke	完全椭圆积分
erf	误差函数
erfc	互补误差函数
erfcx	比例互补误差函数
erfinv	逆误差函数
expint	指数积分函数
gamma	γ 函数
gcd	最大公约数
gammainc	非完全 γ 函数

续表

	lcm	最小公倍数
	log2	分割浮点数
	pow2	比例浮点数
	rat	有理逼近
	rats	有理输出
	cart2pol	变卡笛尔坐标为极坐标
	cart2sph	变卡笛尔坐标为球坐标
	pol2cart	变极坐标为卡笛尔坐标
	sph2cart	变球坐标为卡笛尔坐标

表 A.6 基本矩阵和矩阵操作

■ **基本矩阵**		
	zeros	零矩阵
	ones	全"1"矩阵
	eye	单位矩阵
	rand	均匀分布的随机数矩阵
	randn	正态分布的随机数矩阵
	linspace	线性间隔的向量
	logspace	对数间隔的向量
	meshgrid	三维图形的 X 和 Y 数组
	:	规则间隔的向量
■ **特殊变量和常数**		
	ans	当前的答案
	eps	相对浮点精度
	realmax	最大浮点数
	realmin	最小浮点数
	pi	圆周率值 3.141 592 653 589 7……
	i, j	虚数单位
	inf	无穷大
	nan	非数值
	flops	浮点运算次数
	nargin	函数输入变量数

续表

■ **特殊变量和常数**		
	nargout	函数输出变量数
	computer	计算机类型
	isieee	当计算机采用 IEEE 算术标准时，其值为真
	why	简明的答案
	version	MATLAB 版本号
■ **时间和日期**		
	clock	墙上挂钟
	cputime	CPU 时间(以秒为单位)
	date	日历
	etime	计时函数
	tic	秒表开始执行
	toc	秒表停止
■ **矩阵操作**		
	diag	建立或提取对角阵
	fliplr	矩阵作左右翻转
	flipud	矩阵作上下翻转
	reshape	改变矩阵大小
	rot90	矩阵旋转 90°
	tril	提取矩阵的下三角部分
	triu	提取矩阵的上三角部分
	:	矩阵的索引号，重新排列矩阵

表 A.7　特殊矩阵

	compan	友矩阵
	gallery	几个小的测试矩阵
	hadamard	Hadamard 矩阵
	hankel	Hankel 矩阵
	hilb	Hilbert 矩阵
	invhilb	逆 Hilbert 矩阵
	kron	Kronecker 张量积
	magic	魔方矩阵
	pascal	Pascal 矩阵

续表

	rosser	经典的对称特征值测试问题
	toeplitz	Toeplitz 矩阵
	vander	Vandermonde 矩阵
	wilkinson	Wilkinson 特征值测试矩阵

表 A.8 矩阵函数 —数值线性代数

■ 矩阵分析		
	cond	计算矩阵条件数
	norm	计算矩阵或向量范数
	rcond	Linpack 逆条件值估计
	rank	计算矩阵秩
	det	计算矩阵行列式值
	trace	计算矩阵的迹
	null	零矩阵
	orth	正交化
	rref	减缩行格式矩阵
■ 线性方程		
	\ 和 /	线性方程求解
	chol	Cholesky 分解
	lu	高斯消元法求系数阵
	inv	矩阵求逆
	qr	正交三角矩阵分解(简称 QR 分解)
	qrdelete	从 QR 分解中消去一列
	qrinsert	在 QR 分解中插入一列
	nnls	非负最小二乘
	pinv	矩阵伪逆
	lscov	协方差已知的情况下最小二乘求解
■ 特征值和奇异值		
	eig	求特征值和特征向量
	poly	求特征多项式
	polyeig	多项式特征值问题
	hess	Hessberg 形式

续表

■ 特征值和奇异值		
	qz	广义特征值
	rsf2csf	变实分块对角阵为复对角形式
	cdf2rdf	变复对角矩阵为实分块对角形式
	schur	Schur 分解
	balance	矩阵均衡处理以提高特征值精度
	svd	奇异值分解
■ 矩阵函数		
	expm	矩阵指数
	expm1	实现 expm 的 M 文件
	expm2	通过泰勒级数求矩阵指数
	expm3	通过特征值和特征向量求矩阵指数
	logm	矩阵对数
	sqrtm	矩阵开平方根
	funm	一般矩阵的计算

表 A.9　稀疏矩阵函数

■ 基本稀疏矩阵		
	speye	稀疏单位矩阵
	sprandn	稀疏随机矩阵
	sprandsym	对称的稀疏随机矩阵
	spdiags	从对角阵中形成稀疏矩阵
■ 完全矩阵和稀疏矩阵之间变换		
	sparse	从非零元素及其序号中形成稀疏矩阵
	full	变稀疏矩阵为完全矩阵
	find	找出非零元素的序号
	spconvert	稀疏矩阵外部结构的变换
■ 稀疏矩阵非零元素的处理		
	nnz	非零元素的数目
	nonzeros	非零元素
	nzmax	分配给非零元素的存储量
	spones	用“1”取代非零元素
	spalloc	为非零元素分配内存

续表

■ 稀疏矩阵非零元素的处理		
	issparse	当矩阵为稀疏矩阵时，其值为真
	spfun	只对非零元素取函数
■ 显示稀疏矩阵		
	spy	显示稀疏结构
	gplot	绘图
■ 排序算法		
	colmmd	列最小度
	symmmd	最小对称度
	symrcm	逆 Cathill - Mckee 序
	colperm	基于非零元素按列排序
	randperm	随机排列向量
	dmperm	Dulmage - Mendelsohn 分解
■ 范数、条件数和秩		
	normest	2 范数估计
	condest	1 范数条件估计
	sprank	结构化秩
■ 树型操作		
	treelayout	显示一个或多个结构树
	treeplot	画结构树
	etree	求矩阵的消元树
	etreeplot	画消元树图
■ 其它		
	symbfact	符号分解分析
	spparms	为稀疏矩阵处理过程设置参数
	spaugment	形成最小二乘增广系统

表 A.10 数据分析和傅里叶变换函数

■ 基本操作		
	max	取最大分量
	min	取最小分量
	mean	求均值
	median	求中值

续表

■ 基本操作		
	std	求标准差
	sort	按升序排列
	sum	求各元素之和
	prod	求各元素之积
	cumsum	求元素累积和
	cumprod	求元素累积积
	trapz	利用梯形法计算数值积分
■ 有限差分		
	diff	计算差分和近似微分
	gradient	计算近似梯度
	del2	5 点离散拉普拉斯变换
■ 向量操作		
	cross	向量的矢量积
	dot	向量的点积
■ 相关		
	corrcoef	求相关系数
	cov	求协方差矩阵
	subspace	子空间之间的夹角
■ 滤波和卷积		
	filter	一维数字滤波器
	filter2	二维数字滤波器
	conv	卷积和多项式乘法
	conv2	二维卷积
	deconv	反卷积和多项式除法
■ 傅里叶变换		
	fft	离散傅里叶变换
	fft2	二维离散傅里叶变换
	ifft	离散逆傅里叶变换
	ifft2	二维离散逆傅里叶变换
	abs	取模(绝对值)
	angle	取相角
	unwrap	删除跨越 360°边界的相角
	fftshift	将零点平移到频谱中心

续表

■ 傅里叶变换		
	cplxpair	将数值分类成复共轭对
	nextpow2	最靠近 2 的幂次

表 A.11　泛函——非线性数值方法

	ode23	低阶法求解常微分方程
	ode23p	低阶法求解常微分方程并绘出结果图形
	ode45	高阶法求解常微分方程
	quad	低阶法计算数值积分
	quad8	高阶法计算数值积分
	fmin	单变量函数的极小化
	fmins	多变量函数的极小化
	fzero	找出单变量函数的零点
	fplot	函数绘图

表 A.12　多项式和内插函数

■ 多项式		
	roots	求多项式根
	poly	构造具有指定根的多项式
	polyval	多项式计算
	polyvalm	带矩阵变量的多项式计算
	residue	部分分式展开(留数计算)
	polyfit	数据的多项式拟合
	polyder	微分多项式
	conv	多项式乘法
	deconv	多项式除法
■ 数据内插		
	interp1	一维数据内插(一维查表)
	interp2	二维数据内插(二维查表)
	interpft	利用 FFT 进行一维数据内插
	griddata	数据网格

续表

■ 样条内插		
	spline	3次样条数据内插
	ppval	分段多项式计算

表 A.13 通用图形函数

■ 建立和控制图形窗口		
	figure	建立图形(图形窗口)
	gcf	获取当前图形的句柄
	clf	消除当前图形
	close	关闭图形
■ 建立和控制坐标系		
	subplot	在标定位置上建立坐标系
	axes	在任意位置上建立坐标系
	gca	获取当前坐标系的句柄
	cla	消除当前坐标系
	axis	控制坐标系的刻度和形式
	caxis	控制伪彩色坐标刻度
	hold	保持当前图形
■ 句柄图形对象		
	figure	建立图形窗口
	axes	建立坐标系
	line	建立曲线
	text	建立文本串
	patch	建立图形填充块
	surface	建立曲面
	image	建立图像
	uicontrol	建立用户界面控制
	uimenu	建立用户界面菜单
■ 句柄图形操作		
	set	设置对象特性
	get	获取对象特性
	reset	重置对象特性
	delete	删除对象

续表

■ **句柄图形操作**		
	gco	获取当前对象的句柄
	drawnow	填充未完成绘图事件
	newplot	预测 NextPlot 性质的 M 文件
	findobj	寻找指定特性值的对象
■ **打印和存储**		
	print	打印图形或保存图形
	printopt	配置本地打印机缺省值
	orient	设置纸张取向
	capture	屏幕抓取当前图形
■ **动画**		
	moviein	初始化动画帧内存
	getframe	获取动画帧
	movie	播放所记录的动画帧
■ **其它**		
	ginput	用鼠标输入图形
	ishold	返回 Hold 状态
	graymon	设置灰度显示器的图形缺省值
	rbbox	涂抹块
	rotate	沿指定方向旋转对象
	terminal	设置图形终端类型
	uiputfile	弹出保存文件的对话框
	uigetfile	弹出询问文件名的对话框
	whitebg	设置白色背景的图形窗口缺省值
	zoom	二维图形的放大、缩小
	waitforbuttonpress	在图形中等待按键/按钮
■ **用户界面工具**		
	dialog	主对话框建立 M 文件
	figflag	当图形为当前显示时其值为真
	layout	定义对话框布局参数
	uiguide	关于用户界面约定/标准/建议的说明
■ **对话框**		
	errordlg	建立出错对话框
	helpdlg	建立帮助对话框

续表

■ 对话框		
	questdlg	建立提问对话框
	warndlg	建立警告对话框
■ 打印实用工具		
	prtps	PostScript 打印机驱动程序
	prtwin	MS Windows 驱动程序

表 A. 14　二维图形函数

■ 基本 X－Y 图形		
	plot	线性图形
	loglog	对数坐标图形
	semilogx	半对数坐标图形(X 轴为对数坐标)
	semilogy	半对数坐标图形(Y 轴为对数坐标)
	fill	绘制二维多边形填充图
■ 特殊 X－Y 图形		
	polar	极坐标图
	bar	条形图
	stem	离散序列图或杆图
	stairs	阶梯图
	errorbar	误差条图
	hist	直方图
	rose	角度直方图
	compass	区域图
	feather	箭头图
	fplot	绘图函数
	comet	星点图
■ 图形注释		
	title	图形标题
	xlabel	X 轴标记
	ylabel	Y 轴标记
	text	文本注释
	gtext	用鼠标放置文本
	grid	网格线

表 A.15　三维图形函数

■ 曲线和区域填充命令		
	plot3	在三维空间中绘制曲线和点
	fill3	在三维空间中绘制并填充三维多边形
	comet3	三维星点图
■ 三维数据的等高线和其它二维图形		
	contour	等高线图
	contour3	三维等高线图
	clabel	在等高线图上标注高度
	contourc	等高线图计算
	pcolor	伪彩色图
	quiver	箭头图
■ 曲面和网格图形		
	mesh	三维网格曲面
	meshc	网格和等高线混合图形
	meshz	带零平面的三维网格图
	surf	三维曲面阴影图
	surfc	曲面和等高线混合图形
	surfl	带亮度的三维曲面阴影图
	waterfall	落差图
■ 立体可视化		
	slice	立体可视图
■ 图形外形		
	view	指定三维图形视点
	viewmtx	显示变换矩阵
	hidden	设置网格消隐方式
	shading	彩色阴影方式
	axis	坐标轴刻度和外形
	caxis	伪彩色坐标轴刻度
	colormap	颜色对照表
■ 图形注释		
	title	图形标题

续表

■ 图形注释		
	xlable	X 轴标记
	ylable	Y 轴标记
	zlable	Z 轴标记
	text	文本注释
	gtext	用鼠标放置文本
	grid	网格线
■ 三维对象		
	cylinder	产生圆柱体
	sphere	产生球

表 A.16　语言结构和调试

■ MATLAB 编程语言		
	script	有关 MATLAB 的底稿文件和 M 文件的说明
	function	增加新的函数
	eval	执行由 MATLAB 表达式构成的字串
	feval	执行由字串指定的函数
	global	定义全局变量
	nargchk	有效的输入变量数
	lasterr	保持出错信息
■ 程序控制流		
	if	条件执行语句
	else	与 if 命令配合使用
	elseif	与 if 命令配合使用
	end	for，while 和 if 语句的结束
	for	重复执行指定次数(循环)
	while	重复执行不定次数(循环)
	break	终止循环的执行
	return	返回引用的函数
	error	显示信息并终止函数执行
■ 交互输入		
	input	提示用户输入
	keyboard	像底稿文件一样使用键盘输入

续表

■ **交互输入**		
	menu	产生由用户输入选择的菜单
	pause	等待用户响应
	uimenu	建立用户界面菜单
	uicontrol	建立用户界面控制
■ **调试命令**		
	dbstop	设置断点
	dbclear	删除断点
	dbcont	继续执行
	dbdown	改变局部工作空间的内容
	dbstack	列出调用者
	dbstatus	列出所有断点
	dbstep	执行一条或多条语句
	dbtype	带行号列出 M 文件
	dbup	改变局部工作空间的内容
	dbquit	退出调试状态
	mexdebug	调试 MEX 文件

表 A. 17　颜色控制和亮度模型函数

■ **颜色控制**		
	colormap	颜色对照表
	caxis	伪彩色坐标轴刻度
	shading	彩色阴影方式
■ **颜色板**		
	hsv	色彩　饱和度颜色板
	gray	线性灰度颜色板
	hot	黑　红—黄—白颜色板
	cool	深蓝深红阴影颜色板
	bone	以蓝色为基调的灰度颜色板
	copper	线性青铜色调的颜色板
	pink	线性粉红阴影颜色板
	prism	光谱颜色板
	jet	hsv 颜色板的变型

续表

■ 颜色板		
	flag	红、白、蓝、黑交替的颜色板
■ 颜色板相关函数		
	colorbar	显示颜色条
	hsv2rgb	变 HSV 为 RGB 3 色
	rgb2hsv	变 RGB 为 HSV
	contrast	变灰度颜色板为增强图像对比
	brighten	颜色板加亮或变暗
	spinmap	颜色板旋转
	rgbplot	颜色板绘图
■ 加亮模式		
	surfl	带亮度的三维曲面阴影图
	specular	镜面反射
	diffuse	漫反射
	surfnorm	曲面法线

表 A.18　字符串函数

■ 一般函数		
	strings	MATLAB 中有关字符串函数的说明
	abs	变字符串为数值
	setstr	变数值为字符串
	isstr	当变量为字符串时其值为真
	blanks	空串
	deblank	删除尾部的空串
	str2mat	从各个字符串中形成文本矩阵
	eval	执行由 MATLAB 表达式组成的串
■ 字符串比较		
	strcmp	比较字符串
	findstr	在一字符串中查找另一子串
	upper	变字符串为大写
	lower	变字符串为小写
	isletter	当变量为字母时，其值为真
	isspace	当变量为空白字符时，其值为真

续表

■ **字符串比较**		
	strrep	取代字符串
	strtok	在字符串中查找标记
■ **字符串与数值之间变换**		
	num2str	变数值为字符串
	int2str	变整数为字符串
	str2num	变字符串为数值
	sprintf	变数值为格式控制下的字符串
	sscanf	变字符串为格式控制下的数值
■ **十进制数与十六进制数之间变换**		
	hex2num	变十六进制数为 IEEE 标准下的浮点数
	hex2dec	变十六进制数为十进制数
	dec2hex	变十进制数为十六进制数

表 A.19　音频处理函数

■ **一般音频函数**		
	sound	变向量为音频信号
	saxis	音频轴刻度
■ **特定计算机音频函数**		
	auwrite	写按 Wu - law 编码的音频文件
	auread	读按 Wu - law 编码的音频文件
	wavwrite	写 MS Windows 的.WAV 音频文件
	wavread	读 MS Windows 的.WAV 音频文件
	mu2lin	变 Wu - law 编码音频信号为线性音频信号
	lin2mu	变线性音频信号为 Wu - law 编码音频信号

表 A.20　低级文件 I/O 函数

■ **打开和关闭文件**		
	fopen	打开文件
	fclose	关闭文件
■ **未格式化 I/O**		
	fread	从文件读二进制数据

续表

■ 未格式化 I/O		
	fwrite	二进制数据写入到文件
■ 格式化 I/O		
	fscanf	从文件中读格式化的数据
	fprintf	将格式化数据写入到文件
	fgetl	从文件中读行，并丢弃换行符
	fgets	从文件中读行，并保持换行符
■ 文件定位		
	ferror	查询文件 I/O 出错状态
	feof	测试文件尾
	fseek	设置文件位置指针
	ftell	获取文件位置指针
	frewind	反绕文件
■ 字符串变换		
	sprintf	将格式化数据写到字符串
	sscanf	从格式化字符串中读取
■ 文件 I/O		
	wk1const	WK1 记录定义
	wk1read	读 WK1 文件
	wk1write	在 WK1 格式化文件中写矩阵
	wk1wrec	写 WK1 记录头
	csvread	从逗号间隔的格式化文件中读一矩阵
	csvwrite	写一矩阵到逗号间隔的格式化文件中
	dlmread	从以 ASCII 码限界的文件中读一矩阵
	dlmwrite	按 ASCII 码限界的文件格式写一矩阵

附录 B

Toolbox 函数

MATLAB 系统提供了许多 Toolbox(工具箱),而且由于 MATLAB 的可扩充性,Toolbox 的数目与日俱增,这里仅列出一些基本的工具箱函数,以备用户查阅。

表 B.1 为本附录所列出的 Toolbox。表 B.2~B.11 为各个 Toolbox 所提供的工具函数。

表 B.1　MATLAB 提供的部分 Toolbox

目 录 名	工 具 箱 名 称	索　　引
local	局部函数库	表 B.2
signal	信号处理	表 B.3
image	图像处理	表 B.4
control	控制系统	表 B.5
ncd	非线性控制设计	表 B.6
robust	鲁棒控制	表 B.7
ident	系统辨识	表 B.8
optim	最优化	表 B.9
nnet	神经网络	表 B.10
fuzzy	模糊系统	表 B.11

表 B.2　局部函数库

	matlabrc	MATLAB 的主启动 M 文件
	printopt	设置打印选项

表 B.3　信号处理工具箱

■ 波形产生		
	sawtooth	产生锯齿波或三角波
	square	产生方波
	sinc	产生 sinc 或 $\frac{\sin(\pi t)}{\pi t}$ 函数
	diric	产生 Dirichlet 或周期 sinc 函数

续表

■ 滤波器分析和实现	
abs	取绝对值(幅值)
angle	取相角
conv	求卷积
fftfilt	重叠相加法 FFT 滤波器实现
filter	直接滤波器实现
filtfilt	零相位数字滤波
filtic	filter 函数初始条件选择
freqs	模拟滤波器频率响应
freqspace	频率响应中的频率间隔
freqz	数字滤波器频率响应
grpdelay	平均滤波延迟(群延迟)
impz	数字滤波器的冲激响应
zplane	离散系统零极点图
■ 线性系统变换	
convmtx	卷积矩阵
poly2rc	从多项式系数中计算反射系数
rc2poly	从反射系数中计算多项式系数
residuez	Z 变换部分分式展开或留数计算
sos2ss	变系统二阶分割形式为状态空间形式
sos2tf	变系统二阶分割形式为传递函数形式
sos2zp	变系统二阶分割形式为零极点增益形式
ss2sos	变系统状态空间形式为二阶分割形式
ss2tf	变系统状态空间形式为传递函数形式
ss2zp	变系统状态空间形式为零极点增益形式
tf2ss	变系统传递函数形式为状态空间形式
tf2zp	变系统传递函数形式为零极点增益形式
zp2sos	变系统零极点增益形式为二阶分割形式
zp2ss	变系统零极点增益形式为状态空间形式
zp2tf	变系统零极点增益形式为传递函数形式

续表

■ IIR 滤波器设计	
besself	Bessel(贝塞尔)模拟滤波器设计
butter	Butterworth(比特沃思)滤波器设计
cheby1	Chebyshev(切比雪夫)Ⅰ型滤波器设计
cheby2	Chebyshev(切比雪夫)Ⅱ型滤波器设计
ellip	椭圆滤波器设计
yulewalk	递归数字滤波器设计
■ IIR 滤波器阶的选择	
buttord	Butterworth 滤波器阶的选择
cheb1ord	Chebyshev Ⅰ型滤波器阶的选择
cheb2ord	Chebyshev Ⅱ型滤波器阶的选择
ellipord	椭圆滤波器阶的选择
■ FIR 滤波器设计	
fir1	基于窗函数的 FIR 滤波器设计——标准响应
fir2	基于窗函数的 FIR 滤波器设计——任意响应
firls	最小二乘 FIR 滤波器设计
intfilt	内插 FIR 滤波器设计
remez	Parks-McCellan 最优 FIR 滤波器设计
remezord	Parks-McCellan 最优 FIR 滤波器阶估计
■ 变换	
czt	线性调频 Z 变换
dct	离散余弦变换(DCT)
idct	逆离散余弦变换
dftmtx	离散傅里叶变换矩阵
fft	一维快速傅里叶变换
ifft	一维逆快速傅里叶变换
fftshift	重新排列 FFT 的输出
hilbert	Hilbert(希尔伯特)变换
■ 统计信号处理	
cov	协方差矩阵
xcov	交叉协方差函数估计
corrcoef	相关系数矩阵

续表

■ **统计信号处理**		
	xcorr	互相关函数估计
	cohere	相关函数平方幅值估计
	csd	互谱密度(CSD)估计
	psd	信号功率谱密度(PSD)估计
	tfe	从输入输出中估计传递函数
■ **窗函数**		
	boxcar	矩形窗
	triang	三角窗
	bartlett	Bartlett(巴特利特)窗
	hamming	Hamming(哈明)窗
	hanning	Hanning(汉宁)窗
	blackman	Blackman(布莱克曼)窗
	chebwin	Chebyshev 窗
	kaiser	Kaiser 窗
■ **参数化建模**		
	invfreqs	模拟滤波器拟合频率响应
	invfreqz	离散滤波器拟合频率响应
	prony	利用 Prony 方法的离散滤波器拟合时间响应
	stmcb	利用 Steiglitz - McBride 迭代方法求线性模型
	levinson	Levinson - Durbin 递归算法
	lpc	线性预测系数
■ **特殊操作**		
	rceps	实倒谱和最小相位重构
	cceps	倒谱分析和最小相位重构
	decimate	降低序列的取样速率
	interp	提高取样速率(内插)
	resample	改变取样速率
	medfilt1	一维中值滤波
	deconv	反卷积和多项式除法
	modulate	通讯仿真中的调制
	demod	通讯仿真中的解调

续表

■ 特殊操作		
	vco	电压控制振荡器
	specgram	频谱分析
■ 模拟原型滤波器设计		
	besselap	Bessel 模拟低通滤波器原型
	buttap	Butterworth 模拟低通滤波器原型
	cheb1ap	Chebyshev I 型模拟低通滤波器原型
	cheb2ap	Chebyshev II 型模拟低通滤波器原型
	ellipap	椭圆模拟低通滤波器原型
■ 频率变换		
	lp2bp	低通到带通模拟滤波器变换
	lp2hp	低通到高通模拟滤波器变换
	lp2bs	低通到带阻模拟滤波器变换
	lp2lp	低通到低通模拟滤波器变换
■ 滤波器离散化		
	bilinear	双线性变换
	impinvar	冲激响应不变法实现模拟到数字的滤波器变换
■ 其它		
	conv2	二维卷积
	cplxpair	将复数归成复共轭对
	detrend	删除线性趋势
	fft2	二维快速傅里叶变换
	ifft2	二维逆快速傅里叶变换
	filter2	二维数字滤波器
	polystab	稳定多项式
	xcorr2	二维互相关

表 B.4 图像处理工具箱

■ 图像输入/输出		
	bmpread	从磁盘中读 BMP(Microsoft Windows 下的位图)文件
	bmpwrite	将一 BMP 文件写入到磁盘
	gifread	从磁盘中读 GIF 文件
	gifwrite	将 GIF 文件写入磁盘
	hdfpeek	在 HDF 文件中列出目标标记/参考对
	hdfread	从 HDF 文件中读取数据
	hdwrite	写数据到 HDF 文件中
	pcxread	从磁盘中读 PCX 文件
	pcxwrite	将 PCX 文件写入磁盘
	tiffread	从磁盘中读 TIFF 文件
	tiffwrite	将 TIFF 文件写入磁盘
	xwdread	从磁盘中读 XWD 文件
	xwdwrite	将 XWD 文件写入磁盘
■ 实用程序		
	getimage	从坐标系中读取图像数据
	isbw	当图像为黑白图像时，其值为真
	isgray	当图像为灰度图像时，其值为真
	isind	当图像为加标图像时，其值为真
■ 颜色操作		
	brighten	加亮或增暗一颜色板
	cmunique	寻找唯一的颜色板及相应的图像
	cmpermute	置换颜色板位置
	cmgamma	γ 校正颜色板
	cmgamdef	缺省的 γ 校正表
	dither	Floyd - Steinberg 图像颤抖算法
	hsv2rgb	变 HSV 值为 RGB 颜色空间
	imadjust	调整并增强图像强度
	imapprox	利用更少颜色的图像逼近加标图像

续表

函数	功能
■ **颜色操作**	
ntsc2rgb	变 NTSC 值为 RGB 颜色空间
rgb2gray	变 RGB 值为灰度值
rgb2hsv	变 RGB 值为 HSV 颜色空间
rgb2ntsc	变 RGB 值为 NTSC 颜色空间
rgbplot	绘制 RGB 颜色板分量的图形
■ **几何操作**	
imcrop	修剪图像
imresize	改变图像大小
imrotate	旋转图像
truesize	改变图像大小使之具有实际尺寸
imzoom	放大或缩小图像和二维图形
■ **图像增强/分析**	
brighten	增强成削弱颜色板
grayslice	密度(强度)限幅
histeq	直方图均衡化
imadjust	调整和展宽图像强度
imapprox	利用较少颜色的图像逼近图像
imhist	图像直方图
impixel	一像素点的颜色
improfile	轮廓强度
interp2	二维数据内插
■ **图像统计**	
mean2	矩阵的均值
corr2	二维相关系数
std2	二维标准差
■ **形态操作**	
bwarea	二进制图像中的目标区域
dilate	加浓二进制图像
erode	冲淡二进制图像
edge	边界提取
bweuler	欧拉数
bwmorph	形态算子
bwperim	二进制图像中目标的周围

续表

■ **FIR(有限冲激响应)滤波器设计**		
	fsamp2	通过频率取样的二维 FIR 滤波器设计
	fspecial	特殊的二维滤波器
	ftrans2	通过频率变换的二维 FIR 滤波器设计
	fwind1	使用一维窗函数的 FIR 滤波器设计
	fwind2	使用二维窗函数的 FIR 滤波器设计
	imnoise	图像噪声
■ **频率响应**		
	freqspace	二维频率响应的频率空间
	freqz2	二维频率响应
■ **滤波**		
	colfilt	局部非线性滤波
	conv2	二维卷积
	filter2	二维滤波
	medfilt2	二维中值滤波
	mfilter2	屏蔽滤波
	nlfilter	局部非线性滤波
	wiener2	自适应二维维纳滤波
■ **分块处理**		
	bestblk	分块处理的最佳块大小
	blkproc	按块处理一图像
	col2im	重新排列以形成图像
	colfilt	局部非线性滤波
	im2col	重新排列成列
■ **个别区域**		
	mfilter2	屏蔽滤波
	roipoly	定义感兴趣的多边区域
	roicolor	用颜色定义感兴趣的区域
■ **变换**		
	dct2	二维离散余弦变换
	fft2	二维快速傅里叶变换
	fftshift	零频移到频谱中心
	idct2	二维逆离散余弦变换

续表

■ 变换		
	ifft2	二维逆快速傅里叶变换
	radon	Radon 变换
■ 转换		
	dither	Floyd－Steinberg 图像颤抖
	gray2ind	变灰度图像为附标图像
	hsv2rgb	变 HSV 值为 RGB 值
	im2bw	变图像为黑白图形
	imslice	在图像中获取/置入图像块
	ind2gray	变附标图像为灰度图像
	ind2rgb	变附标图像为 RGB 图像
	mat2gray	变矩阵为(灰度)图像
	ntsc2rgb	变 NTSC 值为 RGB 值
	rgb2gray	变 RGB 图像或值为灰度图像或值
	rgb2hsv	变 RGB 值为 HSV 值
	rgb2ind	变 RGB 图像为附标图像
	rgb2ntsc	变 RGB 值为 NTSC 值
■ 图像显示		
	colorbar	显示颜色条
	colormap	设置或获取颜色查找表
	gray	线性灰度颜色板
	hsv，hot，jet	颜色板
	image	显示附标图像
	imagesc	数据定标并按图像显示
	imcontour	图像等高线
	immovie	制作图像动画
	imshow	显示所有类型的图像数据
	montage	按矩形剪辑方式显示图像
	subimage	显示多个图像
	warp	将图像卷成曲面
■ 演示		
	imdemo	一般图像处理演示
	dctdemo	二维离散余弦变换图像压缩演示
	firdemo	二维 FIR 滤波器演示

续表

■ 演示		
	nlfdemo	二维非线性滤波演示
■ 专用函数		
	cumsum3d	三维矩阵封装成二维矩阵时的累积和
	dct	一维离散余弦变换
	dctmtx2	一元二维离散余弦变换矩阵
	ditherc	图像颤抖的 MEX 文件
	elem3d	三维矩阵封装成二维矩阵的元素位置
	getline	利用橡皮线跟踪鼠标移动
	getpts	利用可视点跟踪鼠标移动
	getrect	利用橡皮矩形跟踪鼠标移动
	gif	压缩 GIF 数据
	hdfreadc	读 HDF 文件的 MEX 文件
	hdfpeekc	搜索 HDF 文件的 MEX 文件
	hdfwc	写 HDF 文件的 MEX 文件
	idct	一维逆离散余弦变换
	im2gray	变图像为灰度
	imhistc	图像直方图计算的 MEX 文件
	ndx3d	三维矩阵封装成二维矩阵的索引
	rgb2im	变 RGB 图像为附标或强度图像
	rle	压缩编码数据
	tiff	压缩 tiff 编码数据
	vmquant	与彩色量化 MEX 文件接口的 M 文件
	waitbar	显示等待条
■ MAT 文件		
	bwmorph. mat	bwmorph. m 文件的查找表
	forest. mat	Carmanah Old Growth Forest 的扫描相片
	mri. mat	人体心脑的磁性共振图像
	trees. mat	树的扫描图像

表 B.5 控制系统工具箱

■ 建模		
	append	追加系统动态特性
	augstate	变量状态作为输出
	blkbuild	从方框图中构造状态空间系统
	cloop	系统的闭环
	connect	方框图建模
	conv	两个多项式的卷积
	destim	从增益矩阵中形成离散状态估计器
	dreg	从增益矩阵中形成离散控制器和估计器
	drmodel	产生随机离散模型
	estim	从增益矩阵中形成连续状态估计器
	feedback	反馈系统连接
	ord2	产生二阶系统的 A、B、C、D
	pade	时延的 Pade 近似
	parallel	并行系统连接
	reg	从增益矩阵中形成连续控制器和估计器
	rmodel	产生随机连续模型
	series	串行系统连接
	ssdelete	从模型中删除输入、输出或状态
	ssselect	从大系统中选择子系统
■ 模型变换		
	c2d	变连续系统为离散系统
	c2dm	利用指定方法变连续为离散系统
	c2dt	带一延时变连续为离散系统
	d2c	变离散为连续系统
	d2cm	利用指定方法变离散为连续系统
	poly	变根值表示为多项式表示
	residue	部分分式展开
	ss2tf	变状态空间表示为传递函数表示
	ss2zp	变状态空间表示为零极点表示
	tf2ss	变传递函数表示为状态空间表示
	tf2zp	变传递函数表示为零极点表示
	zp2tf	变零极点表示为传递函数表示
	zp2ss	变零极点表示为状态空间表示

续表

■ **模型简化**		
	balreal	平衡实现
	dbalreal	离散平衡实现
	dmodred	离散模型降阶
	minreal	最小实现和零极点对消
	modred	模型降阶
■ **模型实现**		
	canon	正则形式
	ctrbf	可控阶梯形
	obsvf	可观阶梯形
	ss2ss	采用相似变换
■ **模型特性**		
	covar	相对于白噪声的连续协方差响应
	ctrb	可控性矩阵
	damp	阻尼系数和固有频率
	dcgain	连续稳态(直流)增益
	dcovar	相对于白噪声的离散协方差响应
	ddamp	离散阻尼系数和固有频率
	ddcgain	离散稳态(直流)增益
	dgram	离散可控性和可观性
	dsort	按幅值排序离散特征值
	eig	特征值和特征向量
	esort	按实部排序连续特征值
	gram	可控性和可观性
	obsv	可观性矩阵
	printsys	按格式显示系统
	roots	多项式之根
	tzero	传递零点
	tzero2	利用随机扰动法传递零点
■ **时域响应**		
	dimpulse	离散时间单位冲激响应
	dinitial	离散时间零输入响应
	dlsim	任意输入下的离散时间仿真
	dstep	离散时间阶跃响应

续表

■ **时域响应**	
filter	单输入单输出 Z 变换仿真
impulse	冲激响应
initial	连续时间零输入响应
lsim	任意输入下的连续时间仿真
ltitr	低级时间响应函数
step	阶跃响应
stepfun	阶跃函数
■ **频域响应**	
bode	Bode(波特)图(频域响应)
dbode	离散 Bode 图
dnichols	离散 Nichols 图
dnyquist	离散 Nyquist 图
dsigma	离散奇异值频域图
fbode	连续系统的快速 Bode 图
freqs	拉普拉斯变换频率响应
freqz	Z 变换频率响应
ltifr	低级频率响应函数
margin	增益和相位裕度
nichols	Nichols 图
ngrid	画 Nichols 图的栅格线
nyquist	Nyquist 图
sigma	奇异值频域图
■ **根轨迹**	
pzmap	零极点图
rlocfind	交互式地确定根轨迹增益
rlocus	画根轨迹
sgrid	在 ω_n, z 网格上画连续根轨迹
zgrid	在 ω_n, z 网格上画离散根轨迹
■ **增益选择**	
acker	单输入单输出极点配置
dlqe	离散线性二次估计器设计
dlqew	离散线性二次估计器设计
dlqr	离散线性二次调节器设计

续表

■ 增益选择	
dlqry	输出加权的离散调节器设计
lqe	线性二次估计器设计
lqed	基于连续代价函数的离散估计器设计
lqe2	利用 Schur 法设计线性二次估计器
lqew	一般线性二次估计器设计
lqr	线性二次调节器设计
lqrd	基于连续代价函数的离散调节器设计
lqry	输出加权的调节器设计
lqr2	利用 Schur 法设计线性二次调节器
place	极点配置
■ 方程求解	
are	代数 Riccati 方程求解
dlyap	离散 Lyapunov 方程求解
lyap	连续 Lyapunov 方程求解
lyap2	利用对角化求解 Lyapunov 方程
■ 演示示例	
ctrldemo	控制工具箱介绍
boildemo	锅炉系统的 LQG 设计
jetdemo	喷气式飞机偏航阻尼的典型设计
diskdemo	硬盘控制器的数字控制
kalmdemo	Kalman 滤波器设计和仿真
■ 实用工具	
abcdchk	检测(A，B，C，D)组的一致性
chop	取 n 个重要的位置
dexresp	离散取样响应函数
dfrqint	离散 Bode 图的自动定范围的算法
dfrqint2	离散 Nyquist 图的自动定范围的算法
dmulresp	离散多变量响应函数
dists1	到直线间的距离
dric	离散 Riccati 方程留数计算
dsigma2	DSIGMA 实用工具函数
dtimvec	离散时间响应的自动定范围算法
exresp	取样响应函数

续表

■ 实用工具		
	freqint	Bode 图的自动定范围算法
	freqint2	Nyquist 图的自动定范围算法
	freqresp	低级频率响应函数
	givens	旋转
	housh	构造 Householder 变换
	imargin	利用内插技术求增益和相位裕度
	lab2ser	变标号为字符串
	mulresp	多变量响应函数
	nargchk	检测 M 文件的变量数
	perpxy	寻找最近的正交点
	poly2str	变多项式为字符串
	printmat	带行列号打印矩阵
	ric	Riccati 方程留数计算
	schord	有序 Schwr 分解
	sigma2	SIGMA 实用工具函数
	tfchk	检测传递函数的一致性
	timvec	连续时间响应的自动定范围算法
	tzreduce	在计算过零点时简化系统
	vsort	匹配两根轨迹的向量

表 B.6 非线性控制设计工具箱

■ 对话框管理		
	coneddlg	管理 NCD 工具箱固定编辑器的对话框
	paramdlg	管理 NCD 优化参数的对话框
	rangedlg	管理坐标系范围的对话框
	refdlg	管理 NCD 参考信号的对话框
	stepdlg	管理 NCD 阶跃响应的对话框
	uncerdlg	管理 NCD 不确定变量的对话框
■ 主要界面		
	contrncd	建立 NCD 固定图形的用户界面控制
	menuncd	建立 NCD 固定图形的用户界面菜单
	ncdblock	包含 NCD 框图的 SIMULINK 系统
	optblock	打开一个 NCD 图形的底稿文件
	optfig	建立一个 NCD 固定图形

续表

■ **主要优化**		
	costfun	NCD 优化的代价函数
	nlinopt	执行优化算法
■ **演示示例**		
	ncddemo	包含所有 NCD 演示示例的 SIMULINK 系统
	ncddemo1	PID 控制器
	ncddemo2	带前馈控制器的 LQR
	ncddemo3	多输入多输出的 PI 控制器
	ncddemo4	倒摆演示
■ **教程**		
	ncdtut1	控制设计示例
	ncdtut2	系统辨识示例
■ **用户界面工具**		
	dialog	上对话框建立 M 文件
	errordlg	建立出错对话框
	figflag	当图形为当前显示在屏幕上时，其值为真
	helpdlg	显示一帮助对话框
	layout	定义对话框布局参数的底稿文件
	questdlg	建立提问对话框
	uiguide	有关用户界面约定/标准/建议的说明
	warndlg	建立警告对话框
■ **演示和教程实用工具**		
	ncd1init	为 ncddemo1 的优化进行设置
	ncd2init	为 ncddemo2 的优化进行设置
	ncd3init	为 ncddemo3 的优化进行设置
	ncd4init	为 ncddemo4 的优化进行设置
	penddata	为 ncdtut2(即倒摆)进行设置
■ **界面实用工具**		
	curobj	提供有关当前点的信息
	dividecb	将固定界分为两部分
	delline	从 NCD 图中删除所有的图
	donep	收回 Close 按钮和菜单
	errorncd	管理 NCD 产生的常见错误，它调用 errordlg(出错对话框)

续表

■ 界面实用工具		
	fillaxes	建立约束边界并进行数据检测
	forceit	在已存在的界限内插入一子集
	keyncd	NCD 按键函数
	loadncd	装入并显示 NCD 数据
	makesurf	建立并限界曲面
	snapncd	以 22.5°间隔排出约束条
	refresho	使约束矩阵与图形一致
	saveload	当文件是从 SelectFile 中选择时，其值为真
	texted	收回 Port 可编辑的文本
	undoncd	放弃上次 NCD 图形用户界面的操作
	updatdlg	更新 NCD 对话框
■ 最优化实用工具		
	convertm	变约束矩阵为最优化格式
	minipars	NCD 最小化分析
	montevar	初始化 Monte Carlo 仿真
	ncdglob	定义 NCD 全局变量
	str2mat2	变一行字符串为多行字符串
■ 帮助文本文件(以.HLP 为扩展名)		
	hotkey	热键帮助
	mainncd	一般 NCD 帮助
	paramdlg	最优化参数对话框的帮助
	readncd	与 README.M 文件内容相同
	stepdlg	阶跃响应对话框的帮助
	uncerdlg	不确定性变量对话框的帮助

表 B.7　鲁棒控制工具箱

■ 可选系统数据结构		
	branch	从树中提取一分支
	graft	在树中增加一分支
	issystem	辨识一系统变量
	istree	辨识一树型变量

续表

■ **可选系统数据结构**		
	mksys	为系统建立树变量
	tree	建立树变量
	vrsys	返回标准系统变量名
■ **建模**		
	augss	系统增广(状态空间模型)
	augtf	系统增广(传递函数模型)
	interc	一般多变量内连系统
■ **模型转换**		
	bilin	多变量双线性变换
	des2ss	利用奇异值分解变系统为状态空间系统
	lftf	线性分式变换
	sectf	扇形变换
	stabproj	稳定和逆稳定映射
	slowfast	慢/快分解
	tfm2ss	变传递函数模型为状态空间模型
■ **实用工具**		
	aresolv	广义连续时间 Riccati 方程求解
	daresolv	广义离散时间 Riccati 方程求解
	riccond	连续时间 Riccati 方程的条件数
	driccond	离散时间 Riccati 方程的条件数
	blkrsch	通过 cschur 得到块有序实 Schur 形式
	cschur	通过复旋转得有序复 Schur 形式
■ **多变量 Bode 图**		
	cgloci	连续特性增益轨迹
	dcgloci	离散特性增益轨迹
	dsigma	离散奇异值 Bode 图
	muopt	具有实/复数混合不确定性系统的 SSV(结构化奇异值)上界
	osborne	通过 Osborne 法求得的 SSV 上界
	perron	计算 Perron 特征值
	psv	Perron 特征结构的 SSV
	sigma	连续奇异值 Bode 图
	ssv	结构化奇异值 Bode 图

续表

■ **因子分解技术**	
iofc	内外因子分解(列类型)
iofr	内外因子分解(行类型)
sfl	左边频谱分解
sfr	右边频谱分解
■ **模型简化方法**	
balmr	截断均衡模型简化
bstschml	相对误差 Schur 模型简化
bstschmr	相对误差 Schur 模型简化
imp2ss	从脉冲响应到状态空间实现
obalreal	有序均衡实现
ohklmr	最优 Hankel 极小化逼近
rschur	Schur 模型简化
■ **鲁棒控制综合方法**	
h2lqg	连续时间 H_2 综合
dh2lqg	离散时间 H_2 综合
hinf	连续时间 H_∞ 综合
dhinf	离散时间 H_∞ 综合
hinfopt	H_∞ 综合的 γ 迭代
normh2	计算 H_2 范数
normhinf	计算 H_∞ 范数
lqg	LQG 最优控制综合
ltru	LQG 闭环传递补偿
ltry	LQG 闭环传递补偿
youla	Youla 参数化
■ **演示示例**	
accdemo	弹簧质量标准问题
dintdemo	双积分器系统的 H_∞ 设计
hinfdemo	飞机或大型空间结构的 H_2 或 H_∞ 设计示例
ltrdemo	LQR/LTR 设计示例：飞机
mudemo	μ 综合示例
mudemo1	μ 综合示例
mrdemo	鲁棒模型简化示例
rctdemo	鲁棒控制工具箱演示——主菜单

表 B.8　系统辨识工具箱

■ 仿真和预测		
	idsim	仿真一给定的系统
	pe	计算预测误差
	poly2th	从给定的多项式中构造 Θ 矩阵
	predict	M 步超前预测
■ 数据处理		
	dtrend	从数据集中删除方位
	idfilt	通过 Butterworth 滤波器对数据进行滤波
■ 非参数化估计		
	covf	估计数据矩阵的协方差矩阵
	cra	相关分析
	etfe	估计经验传递函数并计算周期图
	spa	频谱分析
■ 参数估计		
	ar	利用各种方法的 AR 信号模型
	armax	ARMAX 模型预测误差估计
	arx	ARX 模型的最小二乘估计
	bj	Box - Jenkins 模型的预测误差估计
	canstart	具有初值参数估计的多变量模型
	ivar	时间序列的 AR 部分的仪器Ⅳ估计
	ivx	单输出 ARX 模型的仪器可变估计
	iv4	ARX 模型近似最优的Ⅳ估计
	oe	输出误差模型的预测误差估计
	pem	一般线性模型的预测误差估计
■ 建立模型结构		
	arx2th	ARX 模型的 Θ 格式
	canform	正则形模型结构
	mf2th	将用户定义的模型结构封装入 Θ 模型格式中
	modstruc	在 ms2th 函数中使用的模型结构
	ms2th	将标准状态空间参数封装入 Θ 格式中
	poly2th	从给定多项式中产生 Θ 矩阵

续表

■ 处理模型结构		
	fixpar	在状态空间和 ARX 模型结构中，找出要修正的参数
	sett	在 Θ 结构中设置取样间隔
	thinit	参数的(随机)初始值
	unfixpar	在状态空间和 ARX 模型结构中，放松参数
■ 模型变换		
	th2arx	变 Θ 格式模型为 ARX 模型
	th2ff	求模型的频率响应及标准偏差
	th2par	变 Θ 格式为参数和协方差阵
	th2poly	求给定模型相应的多项式
	th2ss	变 Θ 格式为状态空间表示
	th2tf	变 Θ 格式为传递函数表示
	th2zp	求零极点、静态增益和标准偏差
	thc2thd	变连续时间模型为离散时间模型
	thd2thc	变离散时间模型为连续时间模型
■ 模型表示		
	bodeplot	传递函数的 Bode 图或频谱
	ffplot	频域函数
	idplot	输入——输出数据
	nyqplot	传递函数的 Nyquist 图
	present	屏幕上的参数模型
	zpplot	零点和极点
■ 信息提取		
	getmfth	获取定义模型结构的 M 文件的文件名
	getncap	获取数据点数和参数个数
	getff	选取频率函数
	gett	为某模型获取取样间隔
	getzp	在由 th2zp 函数产生的零极点格式中，提取零点和极点
■ 模型合法化		
	compare	将仿真和预测的输出与测量输出比较
	idsim	仿真一给定的系统
	pe	预测误差

续表

■ 模型合法化		
	predict	M 步超前预测
	resid	计算和测试与某模型相关的留数
■ 估计模型的不确定性		
	idsimsd	在仿真模型响应中说明不确定性
	th2ff	模型频率函数和标准偏差
	th2zp	零点、极点、静态增益及其标准偏差
■ 模型结构选择		
	arxstruc	ARX 模型类的损失函数
	ivstruc	单输出类的输出误差拟合
	selstruc	根据各种准则选择模型结构
	struc	arxstruc 和 ivstruc 的典型结构矩阵
■ 递归参数估计		
	rarx	对 AR 模型递归计算估值
	rarmax	对 ARMAX 模型递归计算估值
	rbj	对 Box - Jenkins 模型递归计算估值
	roe	对输出误差模型递归计算估值
	rpem	对一般模型递归计算估值
	rplr	对一般模型递归计算估值
	segment	分段数据并跟踪快变系统

表 B.9 最优化工具箱

■ 非线性最小化函数		
	attgoal	达到多目标
	constr	约束极小化
	fmin	无约束极小化(标量情况)
	fminu	利用梯度搜索的无约束极小化
	fmins	利用单纯形搜索的无约束极小化
	fsolve	非线性方程求解
	leastsq	非线性最小二乘
	minimax	极小极大求解
	seminf	半定极小化

续表

■ **矩阵问题极小化**		
	lp	线性规划
	qp	二次规划
	nnls	非负最小二乘
■ **控制缺省值和选项**		
	foptions	参数设置
■ **演示**		
	optdemo	演示菜单
	tutdemo	启动教程
	bandemo	香蕉型函数的极小化
	goaldemo	目标达到
	dfildemo	有限精度滤波器设计
	datdemo	数据拟合成曲线
■ **内部使用的实用程序**		
三次内插程序		
	cubic	内插 4 点以找出极大值
	cubici1	内插 2 点和梯度，以估计极小值
	cubici2	内插 3 点和 1 梯度
	cubici3	内插 2 点和梯度，以找出步长和极小值
二次内插程序		
	quad2	内插 3 点以找出极大值
	quadinter	内插 3 点以估计极小值
演示实用程序		
	eigfun	返回分类特征值的函数
	elimone	消去一变量
	filtfun	频率响应和根
	filtfun2	频率响应范数和根
	fitfun	返回拟合数据中的误差范数
	fitfun2	返回拟合数据中的误差矢量
半定实用程序		
	semifun	半定问题转换成约束问题
	findmax	在数据向量中内插极大值

续表

半定实用程序		
	findmax2	在数据矩阵中内插极大值
	v2sort	分类两向量，然后删去丢失的元素
目标达到的实用程序		
	goalfun	目标达到问题转换成约束条件问题
	goalgra	变换目标达到问题中的梯度
测试程序		
	toptim	最优化测试组
	toptimf	最优化测试组的测试函数
	toptimg	最优化测试组的测试函数梯度
其它		
	graderr	用于检查梯度的不一致性
	lsint	初始化最小二乘程序的函数
	optint	初始化无约束极小化程序的函数
	searchq	线性搜索程序

表 B.10 神经网络工具箱

■ 误差分析函数		
	errsurf	计算误差曲面
	plotep	在误差曲面上绘制权和基位置图
	plotes	绘制误差曲面图
■ δ 函数		
	deltalin	对 PURELIN 神经元的 δ 函数
	deltalog	对 LOGSIG 神经元的 δ 函数
	deltatan	对 TANSIG 神经元的 δ 函数
■ 设计		
	solvehop	设计 Hopfield 网络
	solvelin	设计线性网络
	solverb	设计径向基网络
	solverbe	设计精确的径向基网络

续表

■ 初始化		
	initc	竞争层初始化
	initelm	Elman 递归网络初始化
	initff	至多三层的前向网络初始化
	initlin	线性层初始化
	initlvq	LVQ 网络初始化
	initp	感知层初始化
	initsm	自组织映射初始化
	midpoint	产生中点值
	nwlog	对 LOGSIG 神经元产生 Nguyen - Widrow 随机数
	nwtan	对 TANSIG 神经元产生 Nguyen - widrow 随机数
	randnc	产生归一化列随机数
	randnr	产生归一化行随机数
	rands	产生对称随机数
■ 学习规则		
	learnbp	反向演播学习规则
	learnbpm	带预测的反向演播学习规则
	learnh	Hebb 学习规则
	learnhd	退化的 Hebb 学习规则
	learnis	内星学习规则
	learnk	Kohonen 学习规则
	learnlm	Levenberg - Marquardt 学习规则
	learnlvq	学习矢量量化规则
	learnos	外星学习规则
	learnp	感知层学习规则
	learnpn	归一化的感知层学习规则
	learnwh	Widrow - Hoff 学习规则
■ 矩阵		
	combvec	创建所有的矢量集
	delaysig	从信号矩阵中建立退化的信号矩阵
	dist	计算矢量距离

续表

■ 矩阵		
	ind2vec	变下标矢量为稀疏矩阵表示
	normc	归一化矩阵列
	normr	归一化矩阵行
	pnormc	伪归一化矩阵列
	quant	离散化成某数值的整数倍
	sumsqr	平方和
	vect2ind	变稀疏矩阵表示为下标矢量
■ 邻域		
	nbdist	使用矢量距离的邻域阵
	nbgrid	使用栅格距离的邻域阵
	nbman	使用 Manhattan 距离的邻域阵
■ 绘图		
	barerr	每个输出矢量的误差条形图表
	hintonw	绘制权值图
	hintonwb	绘制权值和偏差图
	ploterr	绘出网络误差与时间的关系
	plotes	绘制误差曲面
	plotfa	绘出目标模式及网络函数的逼近
	plotpv	绘出限幅神经元的感知器分类
	plotsm	绘制自组织映射图
	plottr	绘出网络误差记录及自适应学习速率
	plotvec	用不同颜色绘制矢量
■ 仿真		
	simuc	竞争层仿真
	simuelm	Elman 递归网络仿真
	simuff	前向网络仿真
	simuhop	Hopfield 网络仿真
	simulin	线性层仿真
	simup	感知层仿真
	simurb	径向基网络仿真
	simusm	自组织映射仿真

续表

■ 训练		
	trainbp	利用反向演播训练前向网络
	trairbpx	利用快速反向演播训练网络
	trainc	训练竞争层网络
	trainelm	训练 Elman 递归网络
	trainlvq	训练 LVQ 网络
	trainp	利用感知规则训练感知层
	trainpn	利用归一化感知规则训练感知层
	trainsm	利用 Kohonen 规则训练自组织映射
	trainwh	利用 Widrow Hoff 规则训练线性层
■ 传递函数		
	compet	竞争层传递函数
	hardlim	硬限幅传递函数
	hardlims	对称硬限幅传递函数
	logsig	对数 S 型传递函数
	purelin	线性传递函数
	radbas	径向基传递函数
	satlins	对称饱和线性传递函数
	tansig	正切 S 型传递函数

表 B. 11 模糊系统工具箱

■ GUI 编辑器		
	fuzzy	基本 FIS(模糊推理系统)编辑器
	mfedit	隶属度函数编辑器
	ruleedit	规则编辑器及(句法)分析程序
	releview	规则观察器及模糊推理框图
	surfview	输出曲面观测器
■ 隶属度函数		
	dsigmf	两个“S”形隶属度函数的差
	gauss2mf	双边高斯曲线隶属度函数
	gaussmf	高斯曲线隶属度函数
	gbellmf	广义钟形隶属度函数

续表

	pimf	π形隶属度函数
	psigmf	两个"S"形隶属度函数的积
	smf	"S"形隶属度函数
	sigmf	"sigmoid(S)"形隶属度函数
	trapmf	梯形隶属度函数
	trimf	三角形隶属度函数
	zmf	"Z"形隶属度函数
■ 命令行 FIS 函数		
	addmf	将隶属度函数加到 FIS 中
	addrule	将规则加到 FIS 中
	addvar	将变量加到 FIS 中
	defuzz	去模糊隶属度函数
	evalfis	完成模糊推理计算
	evalmf	隶属度函数计算
	gensurf	产生 FIS 输出曲面
	getfis	获得模糊系统的特性
	mf2mf	在函数之间变换参数
	newfis	产生新的 FIS
	parsrule	分析模糊规则
	plotfis	显示 FIS 输入/输出图
	plotmf	显示出一个变量的所有隶属度函数
	readfis	从磁盘中装入 FIS
	rmmf	从 FIS 删除隶属度函数
	rmvar	从 FIS 中删除变量
	setfis	设置模糊系统特性
	showfis	显示带注释的 FIS
	showrule	显示 FIS 规则
	writefis	在磁盘中保存 FIS
■ 先进技术		
	anfis	Sugeno—type FIS 的训练程序
	fcm	利用模糊 C 平均聚集方法找出簇
	genfis1	利用一般方法产生 FIS 矩阵
	genfis2	利用减法聚集方法产生 FIS 矩阵
	subclust	利用减法聚集方法估计簇中心

参 考 文 献

1 楼顺天，于卫，闫华梁编著．MATLAB 程序设计语言．西安：西安电子科技大学出版社，1997

2 MATLAB user's Guide. The Mathworks, Inc. 1995

3 MATLAB Reference Guide. The Mathworks. Inc, 1995

4 SIMULINK User's Guide. The Mathworks, Inc, 1995

5 Control System Toolbox User's Guide. The Mathworks, Inc, 1995

6 System Identification Toolbox User's Guide. The Mathworks, Inc, 1995

7 Ogata K. Designing Linear Control Systems with MATLAB. Prentice－Hall, 1994

8 Moscinski J, etc.. Advanced Control with MATLAB and SIMULINK. Ellis Horwood Limited, 1995

欢迎选购西安电子科技大学出版社各类图书

书名	定价
nternet 中文网站地址簿——精彩中文网址终极推荐	25.00
中文版 Internet Explorer 4.0 套件使用大全	33.00
Intranet Ware 中文版精解	20.50
Intranet 技术及其应用	19.00
MODEM 通信编程技术	20.00
计算机网络技术	36.00
实用网络编程技术	20.50
Windows NT4.0 环境下 Intranet 组建技术	22.00
轻松使用 Microsoft FrontPage 98	9.00
互联网 Internet 和用户软件 Netscape	16.50
NetWare 3.X~4.X 实用培训教程	23.50
电子邮件、Internet 及 WWW 实用技巧	17.60
Internet 操作导航 123 问	13.00
Internet 集成浏览工具	21.00
跟我进入 Internet	22.80
Internet 资源与使用	15.80
Novell 实用网络工程方法	17.50
计算机网络	14.00
中文版 Windows 98 使用指南	26.50
Windows NT Server 4.0(中文版)组网技术	30.00
UCDOS 6.0/7.0 实用操作教程	23.00
DOS 开发环境及其高级技术	45.00
从 DOS 到 Windows	17.50
四通利方 Rich Win 4.2 操作与使用技巧	20.00
中文之星 2.0/2.5 for Windows 95 操作与使用技巧	25.00
中英文 Windows 95 快速通	24.80
中文版 Windows 95 使用大全	36.50
中文 Windows 95 环境与操作	26.00
Windows 95 使用教程	19.80
Windows 95 使用指南	22.00
Windows 3.2 快速入门	9.80
中文 Windows 3.2 实用教程	32.00
Windows 3.2/3.1 简明使用指南	18.00
中文 Windows 3.1、3.2 配置与热点	18.50
UNIX 系统初级教程	23.00
操作系统教程	16.00
Norton 8.0 中文版操作应用技巧	31.80
微型计算机实用反病毒技术指南	16.50
DOS 工具软件例解大全	25.00
Windows 95 多媒体应用程序设计技术	27.00
微机多媒体技术及应用	18.80
多媒体电脑原理、使用与维护	22.50
多媒体电脑安装与测试实用技术	22.80
多媒体技术览要	12.00
全国计算机等级考试(一级)试题分析与应试指南	22.50
全国计算机等级考试(二级)试题分析与应试指南 FoxBASE 语言程序设计	27.00
全国计算机等级考试(二级)试题分析与应试指南 基础部分和 C 语言程序设计	25.00
全国计算机等级考试(三级)模拟试题与解答	19.50
全国计算机等级考试(一级)模拟试题与解答	17.00
计算机等级考试培训教程(一级)	14.80
计算机应用基础教程	21.00
计算机应用办公技能培训教程	12.80
会计电算化实用教程(初级)	28.00
看图学用金蝶财务软件 for Windows	17.50
新编青少年电脑轻松起步教程	23.00
Windows 入门及其文字处理	12.80
计算机原理、操作与文字处理(第二版)	12.80
微机操作与文字处理(修订版)	18.80
WPS 实用指南(2.0~NT1.2)	12.00
家用电脑的选购、使用与维护	14.50
精通电脑 150 问	21.00
电脑使用 300 个怎么办?	28.60
计算机键盘练习与汉字录入技术	5.50
五笔字型速查小字典	4.50
微机常用软件英文提示信息速查手册	16.50
Office 97 中文版快学通	39.00
WPS 97 实用操作教程	20.00
中英文 Word 97 速成教程	20.50
中文 Word 8.0 快学通	20.60
中文 Excel 8.0 快学通	19.80
Photo shop 图像处理软件实用技术	16.00
3D Studio 从入门到精通	24.50
中文版 Microsoft Excel 7.0 实用教程	30.00
汉字 Lotus 1－2－3 R5 for Windows 实用教程	30.00
中文 Power Point 95 快学通	31.50
中文 Excel 95 快学通	31.50
中文 Word 95 快学通	19.50
中文 Word 6.0、7.0 高级技巧	28.00
中文 Word 7.0 使用与提高	29.00

最新中文版 Word 7.0 实用教程	26.00
最新中文 Word 6.0 使用指南	21.00
高级文字处理软件 Word 6.0 应用与开发	21.50
Power Builder 4.0 使用精解	32.00
智能卡技术及应用	9.50
计算机通信技术及其程序设计	22.00
PCI 局部总线开发者指南	8.50
计算机系统安全技术与方法	31.00
自动测试技术与计算器、仪器系统设计	19.50
实用电脑装配与硬件测试技术	21.00
常用电脑传真软硬件的安装与使用	12.80
MD110 程控数字交换机——操作维护教程	20.60
打印机疑难故障诊断与排除	17.00
打印机使用技巧与故障维修 400 例	18.00
计算机通信网原理	16.50
Visual FoxPro 5.0 中文版实用指南	22.00
图解 Visual Foxpro 5.0	23.00
关系数据库 Sybase SQL Server 应用指南	29.00
ORACLE 7 关系数据库实用技术——编程与应用	27.00
Microsoft Visual FoxPro 3.0 使用指南	45.00
Visual FoxPro 3.0b 中文版快学通	31.50
FoxPro 2.6 快速入门	16.00
FoxPro 2.6 实用教程	22.60
FoxPro 2.5、2.6 及其程序设计	28.00
最新 FoxPro 2.6 for Windows 使用详解	20.00
FoxPro 2.5 实用程序设计与技巧	22.00
汉字 FOXBASE+及其程序设计	14.80
汉字 FOXBASE+及其程序设计——习题解答与上机指导	13.40
汉字 FOXBASE+高级程序设计技术——方法、技巧与实例	15.00
JAVA 类库及其实例大全	43.80
Visual C++ 5.0 使用指南	19.50
Borland C++ Builder 使用指南	22.50
Borland C++ Builder 编程技巧与实例	22.00
Power Builder 5.6/6.0 从入门到精通	30.00
跟我学 Quick BASIC	10.50
Visual Basic 5.0 中文版从入门到精通	24.00
Visual Basic 4.0 应用速成	24.00
Visual BASIC 3.0 for Windows 程序设计指南	21.00
Delphi 多媒体程序设计	25.00
Delphi(1.0/2.0)实用编程技术	22.50
Visual Basic 5.0 中文版实用指南	17.00
MATLAB 程序设计语言	16.80
Java 语言及其程序设计	41.00
Borlound C++ 5.0 OWL 5.0 编程技术与实例	35.50
利用 Visual C++ 2.0/4.0 编制 Windows 95 应用程序	29.50
图形用户界面设计与技术——以 Borland C++为工具(含盘)	35.00
Visual C++ for Windows 面向对象程序设计	27.50
微机科学可视化系统设计(含盘)	29.80
C 语言实用软件界面技术	15.00
C 语言实用程序荟萃	18.00
C 程序设计实用教程	14.50
C 语言实践	13.00
TURBO PASCAL 6.0 精讲、题解及应用	20.00
PASCAL 程序设计及其应用	18.00
FORTRAN 语言程序设计(第二版)	16.50
数据结构	10.00
计算方法	8.80
微型计算机原理及应用(本科)	22.00
微型机系统故障分析与实用维修	24.50
微型计算机原理与应用——以 IBM PC 系列机为例	22.00
微型计算机原理与应用(大专)(16 位)	19.00
Motorola 单片机原理及应用技术	20.50
VHDL 硬件描述语言与数字逻辑电路设计	22.00
8098 单片微型计算机应用实例	12.00
IBM PC 微机应用系统设计	14.50
IBM PC 汇编语言程序设计和接口技术	10.00
十六位微型计算机原理及接口技术	15.00
多微处理器系统设计及其实例	12.00
可编程序控制器原理及应用	23.50
计算机控制原理及其应用	23.50
PROTEL 3.31 实用精解	32.50
Auto CAD 高效机械绘图技术	29.00
AutoCAD 12.0 绘图软件包的使用与二次开发技术	27.00
Auto CAD 学与练	16.50
电子 CAD 技术基础	9.50
电子电路 CAD 技术	17.80
电子系统及专用集成电路 CAD 技术	21.50
机械 CAD/CAM 技术概论	11.50
机械 CAD 技术基础	13.20
机械 CAD 应用与开发技术	20.50
孤立子理论及其应用——光孤子理论及光孤子通信	32.80
数据融合理论与应用	20.00

书名	定价
面向对象技术	17.00
神经网络计算	27.00
神经网络应用与实现	29.00
神经网络系统理论	18.50
视觉神经系统与分布式推理理论	12.00
系统核与核度理论及其应用	6.50
实用小波分析	12.00
小波分析及其应用	7.00
UNIX 直通车	16.50
中文 Excel 95 直通车	9.00
C++ 语言直通车	11.50
数码相机的使用	9.80
摄录像技术及多媒体光盘 ----原理、使用与维修	26.50
CD・VCD・DVD----原理、选购、与维修	20.00
VCD・DVD 家庭影院	7.50
视听音响设备----原理・使用・搭配	25.00
家庭影院——组建，使用，维护	18.20
现代家庭视听指南	35.00
现代家用电器——选购、使用与维修大全	26.00
家用微波炉实用技巧	6.00
万用表检修电视机实践指南	
实用电视机维修技巧与方法	21.00
彩电摇控系统、画中画、有线电视加装与维修指南	22.50
电冰箱和冷柜的原理、选用与维修	14.00
空调器及其微电脑控制器的原理与维修	24.80
数字光纤通信设备(上)	19.00
数字光纤通信设备(下)	23.50
电力传动自动控制系统	30.50
集成电路速查大全	24.00
常用办公通信设备的原理、使用与维护——电话机、传真机、传印机、BB 机、大哥大	17.50
ATM 理论及应用	18.50
A/D、D/A 转换器接口技术实用线路	23.50
开关稳压电源——原理、设计与实用电路	25.00
自动控制原理(大专)	10.50
电子线路基础	17.50
录音录像技术	14.80
通讯电子线路	14.00
通信系统原理	22.00
通信基础电源	15.50
纠错码——原理与方法	28.00
电子测量技术基础	17.30
电器原理与技术(电视、录像及家用制冷)	20.00
模拟电子技术	13.40
数字电子技术	14.90
电路分析基础(第二版)	18.00
《电路分析基础》实验与题解	12.00
网络、信号与系统	14.50
电路基础	21.00
电路、信号与系统实验(修订版)	10.50
高频电路原理与分析(第二版)	17.80
电视原理与接收技术	12.80
英汉电脑软件词汇手册	16.80
英汉计算机操作/阅读翻译/应试词汇手册	12.50
电脑英语五周通教程	24.50
高等教育学历文凭基础英语考试指南	15.00
全国职称英语等级考试——模拟测试与阅读辅导(理工类)	20.00
全国职称英语等级考试——模拟测试与阅读辅导(综合类)	21.00
专业英语（第一分册） 中专	16.50
专业英语（第二分册） 中专	16.50
最新考研英语复习指导	17.80
玫瑰花与苹果树(当代英语阅读进阶 Book Ⅰ)	10.20
车轮上的学校（当代英语阅读进阶 Book Ⅱ）	10.00
初雪(当代英语阅读进阶 Book Ⅲ)	9.00
TOEFL 语法满分技巧	26.70
大学英语常见同义词辨析	8.60
大学英语四级结构要点总汇	4.80
大学英语四级阅读综合训练	7.80
大学英语四、六级写作指导	5.00
大学英语六级模拟试题新题型精编	11.00
新编大学英语四级考试模拟题集	10.00
大学英语五、六级研究生英语词汇手册	13.50
大学英语四级考试词汇手册	7.50
新编大学英语四级考试固定词组手册	4.00
科技英语阅读教程	17.00
科技英语语法高级教程	33.50
《线性代数》学习指导与例题分析	11.00
随机过程	13.00
最新考研数学复习指导	20.50
概率论与数理统计	8.00
高等数学(上册)	13.90
高等数学(下册)	13.50
科技英语(电子类)(大专)	14.60
高级操作系统	13.80
计算机操作系统(第二版)	19.60
计算机操作系统(第三版)	27.00

书名	定价
操作系统教程——UNIX 实例分析(第二版)	18.50
UNIX 操作系统教程	16.20
操作系统(修订版)(大专)	14.80
操作系统(中专)(第二版)	10.80
汇编语言程序设计(修订版)	21.50
《汇编语言程序设计》习题解答及实验指导	9.20
单片微机原理与应用(8098)	10.80
单片机原理及应用(51 系列)(中专)	15.50
《单片机原理及应用》学习指导	8.00
微型计算机原理(中专)	18.50
微型计算机原理(16 位机)	24.50
《微型计算机原理》学习指导书	6.50
微型计算机系统设备与维修(中专)	22.75
计算机系统结构(第二版)	19.00
计算机引论(大专)	11.20
软件系统开发技术(第二版)	12.30
数据库原理与应用	12.00
数据库原理及应用(中专)(修订版)	16.80
《数据库原理及应用》实习与实验指导	11.80
PASCAL 程序设计(大专)	12.90
PASCAL 程序设计(大专)	12.80
PASCAL 程序设计(中专)	11.00
计算机绘图	24.50
计算机通信网(修订版)	13.50
编译方法(大专)(修订版)	17.00
计算方法(大专)(修订版)	8.30
离散数学(修订版)	17.80
离散数学(大专)	12.00
管理信息系统概论(大专)	5.60
管理信息系统分析与设计	10.80
音响技术	11.80
控制电机(第二版)	15.00
自动控制基础(修订版)	14.50
工业自动化设备概论	14.00
办公自动化技术与设备	12.00
微机工业控制	12.00
短波通信	14.80
卫星通信(新版)	12.00
图像通信	12.00
扩频通信	9.80
移动通信(新版)	15.00
锁相技术(新版)	11.80
雷达原理	16.80
高频电子线路(中专)(第三版)	17.5
高频电子线路(大专)	13.00
电路分析(大专)	16.50
电工基础(中专)	15.50
数字信号处理	15.00
电视原理与现代电视系统	16.80
电视机原理与技术	15.80
电视接收技术(大专)	11.80
电视原理与接收机(中专)	15.00
天线与电波	13.80
线天线的宽频带技术	9.70
微波技术基础	15.00
电磁场有限元方法	26.80
电磁场理论基础	17.00
电磁场微波技术与天线	14.80
几何绕射理论(新版)	9.65
机构精确度	7.90
电子精密机械导论	10.90
电子机械计算机辅助设计	9.50
电子机械制造工艺学(中专)	10.90
电子工程制图(含习题集)(中专)	24.00
工程制图(含习题集)	33.20
机械基础(中专)	8.90
机械制造——实习教材(中专)	14.50
机械原理与机械零件习题册(中专)	14.75
金属切削机床(中专)	10.50
塑料模设计(中专)	16.00
模具设计与制造	16.00
工模具制造工艺学(中专)	14.70
工业企业经济活动分析(中专)	12.40
工业企业管理(中专)(修订版)	9.50
管理数学(中专)(修订版)	14.80
冲压塑压设备概论(中专)	8.00
管理心理学(中专)	7.90
公共关系学(中专)	8.20

欢迎来函索取本社最新书目和教材介绍，欢迎投稿！

从邮局或银行汇款邮购者， 款到后五天内我社将挂号发书， 加收 15%的包装邮寄费。

通信地址： 西安市太白南路 2 号　西安电子科技大学出版社发行部　邮　编： 710071

电　话：(029)8227828、8202945　传　真：(029)8213675